Adminstration & User Manual SyncManager

Version 2.2.1-41100

Date 9-9-2009

realworldsystems

Realworld Systems
Culemborg
The Netherlands

The information in this document is subject to change without notice and does not represent a commitment on the part of Realworld Systems BV. The software described in this document is subject to a License Agreement and may not be copied to other media except as specifically allowed in the License Agreement. No part of the publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Realworld Systems BV.
All product and company names are ™ or ® trade marks of their respective owners.
Help & Manual is a trade mark of EC Software.
Smallworld is a trade mark of General Electric Company.
Oracle, Oracle Spatial, and Oracle Express are trade marks of Oracle Corporation.
Java is a trademark of Sun Microsystems, Inc.
Windows is a trade mark of Microsoft Corporation.
ArcGis, ArcIMS, ArcWeb, ArcSDE, ArcEngine, ArcObjects, ArcMap, ArcGIS Server and ArcInfo are trademarks or registered trademarks of ESRI.

© Realworld Systems, 2009
All rights reserved.

Table of Contents

1 Realworld SyncManager 6
1.1 Release Information 7
1.2 Preface 9
1.3 Requirements 10
2 Technical overview 13
2.1 Magik Components 13
2.2 Java Components and the Configuration GUI 14
2.3 The Connector 14
3 Installation Guide 16
3.1 Installing SyncManager 16
Windows 16
UNIX 17
3.2 Configuring SyncManager 17
SyncManager Data Store 17
Environment settings 18
Default merge behavior 18
Internal world owners list 19
Building and saving the image 20
3.3 Uninstalling SyncManager 20
4 User Guide 23
4.1 Typical workflows 23
First time synchronization 24
Changing synchronization 24
4.2 The Mapping Editor 25
Starting the Mapping Editor 25
Preferences menu 26
Global preferences 27
Smallworld preferences 29
Oracle preferences 30
Contextual Menu 31
Enumerator Mapping page 32
Network Mapping page 33
Map Types page 34
Mappings page 35
Mappings Explorer 36
Table Info page 37
Column Mappings page 38
Simulate Geometry Mapping dialog 38
Column Mapping info page 40
Join Info page 42
4.3 Mapping Configurations 43
Creating configurations 43
Defining Mappings 44

Updating configurations 46
Synchronize with Data Dictionary 47
Add new collection 47
Add new field 49
Remove field 49
Remove collection 49
Re-export configuration 50
4.4 Synchronization 51
Start synchronizing 51
Analyzing and fixing errors 51
5 Appendixes 55
5.1 Command line utility for indexes 56
5.2 SyncManager metadata 57
syncmanager.ds 57
Oracle tables 59
5.3 Error descriptions 62
5.4 Known limitations 64
6 Release notes 66

Part I

1 Realworld SyncManager

Data synchronization between different environments, like the Smallworld Geographic Information System and Oracle Spatial, is inherently difficult. In order to achieve a meaningful synchronization many differences need to be consolidated. Various synchronization tools, like Smallworld *InSync*, are designed to fill a specific need and often know limitations.

Realworld *SyncManager* offers a more complete approach. It optimizes and extends the existing Smallworld InSync functionality. With an emphasis on data quality it addresses the various differences between the Smallworld and Oracle environments. *SyncManager* comes with an easy to use GUI, is fast, and offers data synchronization in an open environment.

This document contains a brief Technical overview 13, the Installation Guide 16 and the User Guide 23 for the *SyncManager* product.

1.1 Release Information

This section contains information on the version of the *SyncManager* product and the product documentation.

Product version

This document applies to version 2.2.1-41100 of the product. The manual is also compatible with SyncManager versions 2.2.1-40000 and 2.2.1-41000, which are compatible with Smallworld Core Spatial Technology versions 4.0.0 and 4.1.0 respectively.

Changes between subsequent releases are documented in the separate section Release Notes 66.

Document version

Current version: *Version 2.2.1-41100-A.*

23 Jul 2007	Version 1.0.0	- Initial document
16 Oct 2008	Version 1.1.6	- Revised and upgraded to the new version. - SyncManager installation and user guides by Dirk-Jan Bilderbeek - Quality control: Paul Meester, Gerrit Bruins, Eduard Baraitaru, Heiko Heijenga.
20 Feb 2009	Version 2.0.M3-1	- Upgraded document, to fit in new format for product documentation, revised parts of the text for clarity. - Added the section Technical overview 13 - Added Release Notes derived from the pre release notes document.
4 Mar 2009	Version 2.0.M3-2	- Revised almost whole manual, especially the description of the Mapping Editor. This version is the basis of a review on the completeness and correctness of the content. - Context numbers are assigned to all topics
18 Mar 2009	Version 2.0.M3-3	- Created all questions and remarks as yellow painted texts, marked all those topics with topic status "Needs review."
15 May 2009	Version 2.2.0-41000-A	- Updated documentation and solved previously marked texts and topics. - Some open issues to text and release notes
18 May 2009	Version 2.2.0-41000-B	- Completed release notes, but for some more information is required.
20 May 2009	Version 2.2.0-41000-C	- Updated descriptions of the release notes. Marked the Technical Overview as outdated.
25 May 2009	Version 2.2.0-41000-D	- Added the recommendation of 1GB internal memory for SyncManager
10 Jun 2009		- Added the description and solution of the possible error "ORA-01086: savepoint '<savepoint_name>' never established" See here 62.
19 Jun 2009		- Minor reformatting pictures and text; generated Browserhelp ouput
29 Jun 2009		- Added picture of Datamodel menu in Updating configurations 46 - Added description of the Re-export configuration

		50 command - The section Updating Configurations 46 is splitted into subtopics by command. - Added the description of fields in the Simulate Geometry Mapping dialog 38 - Added unresolved issue SYM-3075 - Added more detailed description of the "Switching Full Sync sets Truncate to ON" option in the Contextual Menu 31.
13,21 Aug 09	Version 2.2.0-41000-E	- Updated the Technical overview description
04 Sep 2009	Version 2.2.1-41100-A	- First draft of the Manual compatible to version 2.2.1-41100. (Added also some info about the versioning in the Preface 9)
09 Sep 2009	Version 2.2.1-41100	Final version with updated sections of the SyncManager metadata tables 57 in both the syncmanager.ds and the metadata tables in Oracle.

1.2 Preface

Context

Realworld *SyncManager* provides a solution for the synchronization between data held in Smallworld datastores and Oracle. SyncManager is the only product that offers complete functionality for this.

Audience and required skills

The individual assigned the role of a system administrator for the Smallworld environment should be able to install the Realworld Systems *SyncManager* product. The installation process involves installing the software, configuring the environment, building and saving a session image, and, optionally, creating scripts or user interface elements.

The user guide part of this documentation requires expertise on the level of database administrator and application developer. As *SyncManager* maps (parts of) the data model in the Smallworld environment to a data model in the Oracle environment, expert knowledge of both environments is required.

Because the environment in which *SyncManager* will be used, affect the installation choices, we recommend that a team of all roles involved plan and implement the SyncManager product.

Versioning scheme

SyncManager is a complex product with multiple interactive components (see section Technical Overview 13). It uses a version scheme based on the java components and the versions of Smallworld's Core Spatial Technology, e.g. 2.2.1-41100. The first three digits reflects both the commercial product identification as well as the java components. It than is postfixed with the Realworld Software Modules versioning: the first three digits is the version of CST it is compatible with, the last two the subversion of the realworld smallworld software module roos_sync_manager.
More detailed technical version is available in the file Release.xml that is part of the product.

Distribution forms

The *Realworld Syncmanager* product is available as a standard Realworld software delivery.

The documentation is available as PDF and a compiled HTML library. A future release of the product will make this Help available from the product's Help buttons.

Hardware and Software Requirements

For a complete description of the requirements of this product, see the section Requirements 10.

Author

Realworld Systems
Dirk-Jan Bilderbeek, Eduard Baraitaru, André van den Boogert, Adrian Lici

Address

Venusstraat 17
NL 4105 JH Culemborg
The Netherlands
Phone +31 (0)345 614406
Fax +31 (0)345 614319

Internet www.realworld-systems.com

Support

If you have questions or problems with the installation, configuration and use of our product, you can contact our support department by e-mail support@realworld-systems.com.

1.3 Requirements

- To use *SyncManager*, you need a working installation that contains the following **components:**

  - Smallworld Spatial Core Technology version 4.1.x
  - Oracle Database 10g release 2 (10.2)
  - Java 2 Platform Standard Edition 5.0 (included in the installation)

  Note: 1) It is possible to use Oracle versions from version 9.2, with or without Oracle Spatial, or even Oracle Express. When using any of these versions, you will however not be able to use the full functionality of *SyncManager*. Please contact your Realworld representative to discuss the implications if you need to use any version other than Oracle 10g (10.2).
  2) When you use your own version of Java, you should start it in server mode. So, either through server-class detection, or using the –server option, you should launch the Java HotSpot Server Virtual Machine (server VM) instead of the Java HotSpot Client Virtual Machine (client VM)

- To run *SyncManager*, the following **system resources** should be available:

  - A minimum of 512 MB of available RAM. The recommended value however is 1024 MB.
  - A small amount free disk space for the mappings repository. Mapping files (configurations) are small, but you may create many of them over time.
  - A larger amount of disk space for the SyncManager Data Store. In testing sometimes up to 200 MB was required when a large number of synchronization errors were produced.

- In the Oracle environment you need to create a user or schema for each mapping configuration (see Mapping configurations 43). Create these with at least the following rights:

  CREATE TABLE
  CREATE SESSION
  CREATE PROCEDURE
  CREATE VIEW
  CREATE SEQUENCE
  CREATE TYPE
  UNLIMITED TABLESPACE

  Furthermore the size of the UNDO table space in Oracle should be checked. The initial replication of data is contained in one transaction per collection. The subsequent synchronization updates are contained in one transaction. The size of these transactions may thus be equal to the size of your whole dataset in Smallworld. The size of the UNDO table space in Oracle, in conjunction with its automatic increment, should be large enough to handle these transactions.

- *SyncManager* itself requires about 100 MB of free disk space.

- Tangent point arcs, as used in Smallworld 3.0 and older versions, will be transferred as splines which may not depict the original arc accurately. We recommend that you convert all tangent point arcs to center point arcs.

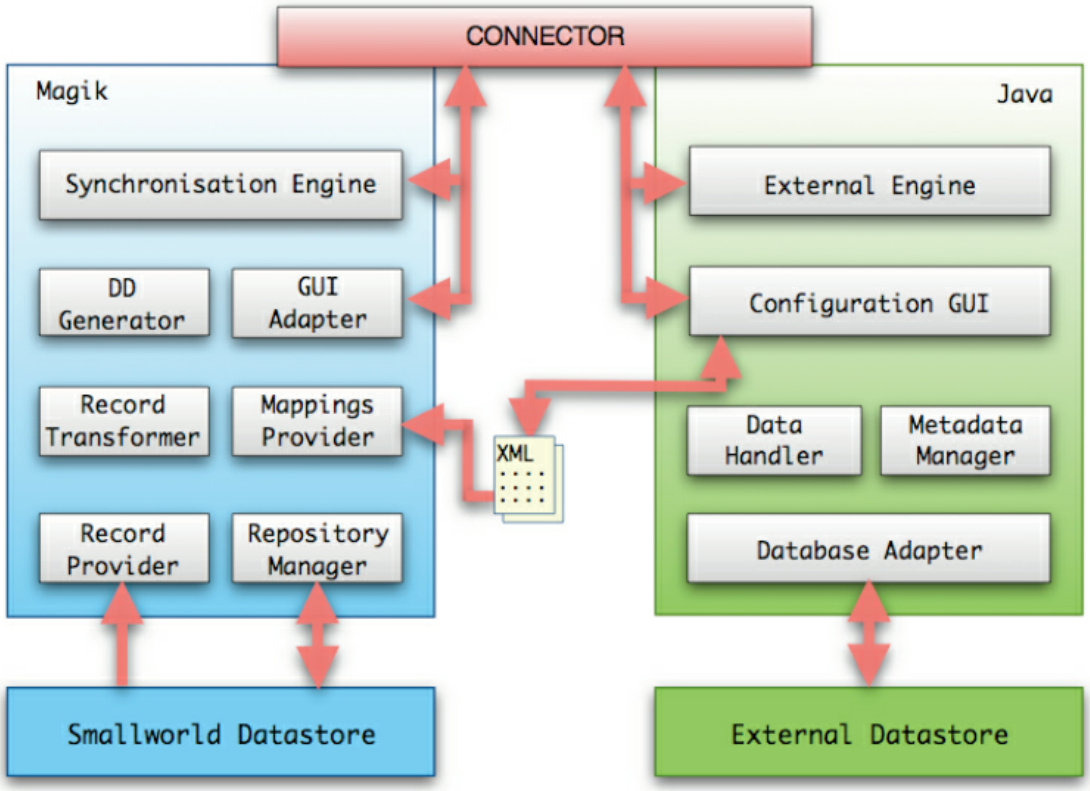
Note: 1) Text join fields are exported with a maximum of 4000 character strings. (See also the section Known limitations 64)
2) Due to the particular way Java handles Float values, all single precision floating point values are processed as double precision floating point values to avoid any loss of information.
3) To enable the synchronization of points with an orientation, *SyncManager* uses the Oracle **SDO_Ordinates** array, rather than the table **SDO_POINT**.

Part II

2 Technical overview

This section provides an overview of the technical structure of *SyncManager*.

Below you can see a simplified diagram of the SyncManager architecture with some communication paths between its components. It contains the following major components: Magik Components 13, Java Components and the Configuration GUI 14 and the Connector between the above major components 14.



SyncManager Architecture - simplified diagram

2.1 Magik Components

The main responsibility of the Magik set of components is to extract from the Smallworld datastore the data to be stored in the external datastore. It consists of the following components:

Synchronisation Engine
The main component that makes sure the actions and data exchanges occur as designed.

Mappings Provider
The component specialized in parsing the mappings configuration file and supplying the engine

with mapping records that detail how Smallworld collections and their fields are mapped to which Oracle tables and columns.

Repository Manager
The component that manages the access to SyncManager Metadata repository.

Record Provider
Responsible for extracting the needed records from the database for a given mapping. When synchronizing data, this component returns only the records that changed since the last export.

Record Transformer
Responsible for changing the shape of the database records from a datastore shape to the shape understood by the Java components.

GUI Adapter
The component that interfaces with the Java GUI and allows the communication to take place. Together with the DD Generator, it provides Java GUI with enough information so that the user can create and manage mapping records.

2.2 Java Components and the Configuration GUI

The main responsibility of the Java set of components is to retrieve the data from the Smallworld session, transform it to a shape suitable for storing in an external database and store the data in the database.

The configuration GUI runs as a Java process outside a Smallworld session. Its purpose is to facilitate creation and the maintenance of the mappings between the Smallworld and the External Database.

2.3 The Connector

The Connector is designed to act as a channel between the two sets of components, Magik and Java, facilitating the communication between them.

Part III

3 Installation Guide

SyncManager is a complex product containing components that run inside a Smallworld session, and components that run as stand-alone processes. You will find however that *SyncManager* is structured, installed, and configured as a layered product in the Smallworld environment.

Follow the steps below to install and configure the product:

1. Check the requirements 10
2. Install the product 16
3. Configure Syncmanager 17

To uninstall the product, read the section Uninstalling SyncManager 20.

3.1 Installing SyncManager

Installing *SyncManager* is a straightforward process. All you need to do is, with the help of an installation wizard or script, copy a number of files to an installation folder (directory) on your system.

Because *SyncManager* is structured as a layered Smallworld product, we recommend you to install *SyncManager* in a separate folder in a Smallworld Core installation folder.

After the installation you must configure the product by hand (see Configuring SyncManager 17).

3.1.1 Windows

The installation procedure on the Windows platform uses a setup wizard to unpack and put all the files in the right places. No configuration takes place during this process.

Follow the next steps:

1. Run the **Setup.exe** found on the installation media (in the Windows folder).
2. A pop-up dialog appears introducing the setup wizard for installing *SyncManager*. This dialog contains a recommendation to close all running applications which you may safely ignore.
   Select **Next** to continue.
3. The following dialog prompts you for a root folder to install *SyncManager* in. Follow the onscreen instructions to specify this folder.
   Typically you select **Browse** to locate the Smallworld Core folder, and then select **Make new folder** to create and specify the *SyncManager* installation folder.
4. Select **Install** to start installing the SyncManager files.
5. Select **Next**, and then **Finish** to complete the installation and to exit the setup program.

After unpacking the files, configure the product as described in Configuring SyncManager 17.

3.1.2 UNIX

On UNIX you need to unpack the distribution archive containing *SyncManager*. This archive and an installation script can be found in the UNIX folder on the distribution media.

The installation script **Install.sh** will prompt you for the name of the installation folder and unpack the archive to the destination given. Alternatively you may unpack the archive yourself to your designated installation folder.

We advise you to install *SyncManager* in a separate directory in the main Smallworld Core installation directory.

After unpacking the files, configure the product as described in Configuring SyncManager 17.

3.2 Configuring SyncManager

After you have installed *SyncManager*, you need to configure the product. The configuration steps are:

1. Installing the SyncManager Data Store 17
2. Configure the SyncManager environment 18
3. (If required) modify the default merge behavior 18
4. (If required) define the world owners list 19
5. Build and save the Smallworld image 20

All relevant environment settings have a default including the variable **SYNC_HOME**. This environment variable should be set to point to the folder:

<syncmanagerInstallDir>\roos_sync_manager

After you have configured the product, it is ready for use. Read the User Guide 23 for information on using *SyncManager*.

3.2.1 SyncManager Data Store

SyncManager uses a Smallworld datastore file to store the product's metadata 57. This **SyncManager.ds** file should be placed somewhere where it may grow (see Requirements 10).

Follow the steps below to install the *SyncManager* datastore:

1. Copy the **syncmanager.ds** file to your chosen location. The file can be found in the **sync_manager_src** module, in the following location:

<syncmanagerInstallDir>\roos_sync_manager\modules\src\sync_manager_src\resources\base\data

2. Set the environment variable **SYNC_MANAGER_DB_DIR** to point to the folder (directory) in which you copied the **syncmanager.ds** file.
   You can put this variable in the environment.bat file used by your installation.

Note: If the variable **SYNC_MANAGER_DB_DIR** is not defined, *SyncManager* assumes that the file resides in the **ds_admin** folder.

3.2.2 Environment settings

The configuration settings of *SyncManager* are defined in the file **sync_manager_configuration.xml**. This file can be found in the folder:

<syncmanagerInstallDir>\roos_sync_manager\modules\src\sync_manager_src\resources\base\data

You should at least check or set the following attributes to reflect your environment:

1. The *Mappings_repository*. In the path attribute of this element you should specify the location where the *mapping files* are to be stored. If a relative path is used, the full path will be computed relative to the location of the XML configuration file. See also chapter Requirements 10.
2. The *Logs_repository*. In the path attribute of this element you should specify the location where the log files are to be stored.
   Various options for the log files are defined in the file: log4j.properties, located in the same folder as the configuration file.
3. *JRE*. The path attribute of this element should point to the location of the Java executable. If this element is not set, or set wrongly, the Java version of the Smallworld Core product is used.

You should also define the environment variable **SYNC_HOME**, which value should point to the path of the **roos_sync_manager** module.

Note:
1) *SyncManager* requires Java 2 Edition 6.0 which is included in the installation.
2) If you set a relative path in any of these elements, the path is considered relative to the location of the configuration file.

3.2.3 Default merge behavior

In order not to limit access to the dataset by other users, *SyncManager* is designed to synchronize or replicate from any alternative except the top alternative. Often this will be a dedicated child alternative to the top alternative, but any alternative that fits the role may be specified in the configuration (see also the User Guide 23).

Using a child alternative however requires a merge of changes in the parent alternative to the alternative used by *SyncManager*. If you require any conflicts that may arise during this merge operation to be handled in a special way, you may change the code that controls the merge.

This code can be found in the file **sync_procs.magik**, located in the folder (directory):

<syncmanagerInstallDir>\roos_sync_manager\modules\src\sync_manager_src\source

3.2.4 Internal world owners list

Smallworld Core Technology allows an RWO (user object) to own an internal world. An application may retrieve the internal world owned by an RWO using the method **world** of **world_record**.

In case an RWO has multiple internal worlds, or if an application uses something other than the world method, you have to configure *SyncManager* with information on how to retrieve these internal world(s). Initially the list is empty. When filled, for each combination, view_name and collection_name an element must be added in the list. Each element is a property list keyed on the universe id, in which the internal worlds exist, containing the methods that must be invoked to retrieve the internal worlds.

To configure *SyncManager* to retrieve the internal world owners list, define a shared constant as shown by the template and example below.

Template:

```
sync_manager.define_shared_constant(:sync_world_owners,
     _block
             _local l_dkt << sw:dual_key_a_table.new()
             # for each pair of (world, RWO) define
             # a propertylist containing pairs of
             # (universe id, retrieve method) for all
             # internal worlds
             >> l_dkt
     _endblock,
     :public)
$
```

Example:

Suppose you have a GIS view called **electricity** in which a user object **substation** is defined. This substation object has internal worlds in both the internals universe (ID 3), and in the schematic universe (ID 2). The methods defined to retrieve these internal worlds are: **:|world|**, and **:|schematic_world()|** respectively.

The code to configure SyncManager could then be as follows.

```
sync_manager.define_shared_constant(:sync_world_owners,
_block
     _local l_dkt << sw:dual_key_a_table.new()
     l_dkt[:electricity,:substation]  << sw:property_list.new_with(
                                         3, :|world|,
                                         2, :|schematic_world()|)
     >> l_dkt
_endblock,
:public)
$
```

Note: When you define this shared constant, *SyncManager will* only map the internal worlds provided by this constant. The standard **world_record** method **world** will be ignored by *SyncManager* when you define this constant.
So, by *not* specifying the **(3, :|world|)** property in the example above, *SyncManager* will not not retrieve the internals world and they will not be mapped.

3.2.5 Building and saving the image

SyncManager is delivered as a layered product for the Smallworld Core Technology platform. This implies that you have to add *SyncManager* as product to a session and load its modules. Typically you would use the Smallworld **Module Manager** to add *SyncManager* to a session.

Finally you probably want to save a session image that contains *SyncManager*.

The two main functions of *SyncManager* are:

- creating mapping configurations with the Mapping Editor, and
- the actual synchronization (replication) of the data.

There are many ways in which you can integrate these functions in an existing application. Details on how to achieve this are however beyond the scope of this manual. For this we have to refer you to the Smallworld documentation.

Below are two code snippets that can be entered at the Magik prompt to start the Mapping Editor, respectively the synchronization of the data.

Code snippet 1: starting the Mapping Editor:

```
_block
     my_sm << sync_manager.new()
     my_sm.open()
_endblock
$
```

Where appropriate you should use **sync_open(_true)** to run the Mapping Editor in a modal window (blocking the calling thread until it exits).

Code snippet 2: starting synchronization:

```
_block
     my_sm << sync_manager.new()
     my_sm.run(<<configuration name>>)
_endblock
$
```

Details of the synchronization process and the log files involved can be found in Synchronization 51.

3.3 Uninstalling SyncManager

The majority of the *SyncManager* software is located in its installation folder (directory). Removing this folder and its contents effectively uninstalls *SyncManager*.

But before removing this folder you may want to reread chapter Configuring SyncManager 17. Using this chapter you may identify all your installation / configuration choices and remove all items created outside the SyncManager installation folder.

Also you may have created one or more specific users/schemas in the Oracle environment which you no longer need. Also the Oracle database may contain tables and data associated with these users/schemas which are no longer relevant. We recommend that, if required, you remove all this by hand, using the appropriate tools in your Oracle environment.

Finally you may want to change or remove the image used in Smallworld, and any scheduled tasks used for synchronizing data.

Part IV

4 User Guide

In order to achieve data synchronization between your Smallworld and Oracle environments with *SyncManager*, you first need to create a *mapping configuration*. This mapping configuration defines which data collections are to be synchronized, and how the inherent differences between the Smallworld and the Oracle environments are to be consolidated.

The two basic steps are:

1. Create (change) a mapping configuration (Mapping configurations 43) with the Mapping Editor 25.
2. Synchronize the data (Synchronization 51)

Please be aware that a mapping configuration links your Smallworld environment to your Oracle environment.

When working with configurations you should always start at the Smallworld side and work towards the Oracle side. The general workflow for mapping configurations is:

1. Update an existing configuration with changes in the Smallworld environment (Updating configurations 46)
2. Create or change mappings, preferences or other information (Defining the mappings 31)
3. Create or update the mapped data model in Oracle (Updating configurations 46)

To explain how these principles work out in practice, we present in the section Synchronization examples 23 two typical workflows in use of *SyncManager*:

1. Initial synchronization 24, and
2. Changed synchronization 24

4.1 Typical workflows

This section contains two typical examples of Synchronization:

1. First time synchronization 24
2. Changing a synchronization 24

4.1.1 First time synchronization

The steps below are the typical workflow when you need to synchronize (part of) your Smallworld data for the first time. A number at the end of a step points to the chapter in the installation or user guide where you can find more detailed information.

1. Create/identify an alternative in Smallworld to synchronize from (Default merge behavior 18)
2. Create a user/schema in Oracle to synchronize to (c-c)
3. Start the Mapping Editor (Running the Mapping Editor 25)
4. Create a new configuration in the Mapping Editor (Creating configurations 43)
5. Specify options and define mappings: (Defining the mappings 31):
   - Configuration options (Configuration details page 26)
   - Mappings (Mappings page 35)
   - Enumerators (Enumerators page 32)
   - Networks (Networks page 33)
   - Used coordinate systems (Coordinate systems page 34)
6. Create the data model in the Oracle database
7. Save the configuration; exit the Mapping Editor
8. Synchronize the data (SyncManager will replicate the first time) (Starting synchronization 51)
9. Optionally create spatial indexes by hand (Appendixes 55)
10. Analyze the results (Analyzing and fixing errors 51)

4.1.2 Changing synchronization

The steps below are the typical workflow when either relevant parts of your Smallworld environment have changed, or when you need to change an existing configuration. A number at the end of a step points to the chapter in the installation or user guide where you can find more detailed information.

1. Start the Mapping Editor (Starting the Mapping Editor 25)
2. Open an existing configuration in the Mapping Editor (Creating configurations 43)
3. If required, update the mapping configuration (Updating configurations (Mapping menu) 46)
4. If required, change options and mappings: (Defining the mappings 31)
   - Configuration options (Configuration details page 26)
   - Mappings (Mappings page 35)
   - Enumerators (Enumerators page 32)
   - Networks (Networks page 33)
   - Used coordinate systems (Coordinate systems page 34)
5. Update the data model in the Oracle database (See Updating configurations 46)
6. Save the configuration; exit the Mapping Editor
7. Synchronize the data (Starting synchronization 51)
8. Optionally create spatial indexes by hand (Appendixes 55)
9. Analyze the results (Analyzing and fixing errors 51)

4.2 The Mapping Editor

The Mapping Editor allows you to create, modify and file Mapping configurations.

The interface is divided into four sub windows:

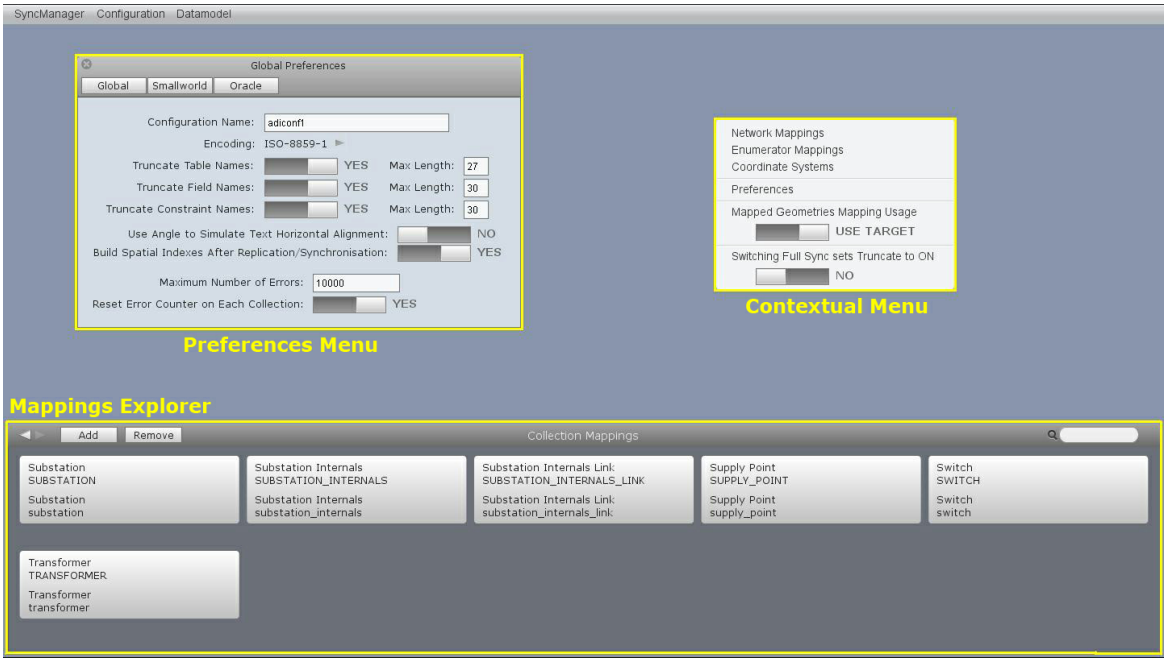
1. The *menubar* with the entries:

   Help: Opening the about window, showing the legend, the help window and command to exit the Mapping Editor

   Configuration: Providing the commands for creating, opening, saving, and closing configurations

   Datamodel: Options for dealing with datamodel changes and Mapping configurations (see Updating configurations 46.)
2. Preferences menu 26 with all the preferences for a configuration.
3. Contextual menu 31 which allows the viewing and editing of Mappings, Coordinate Systems and Preferences
4. Mappings explorer 36 to browse to the table mappings of a configuration

The next sections describe how to use the Mapping Editor.



The Mapping Editor

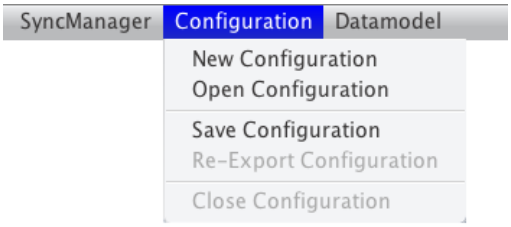
4.2.1 Starting the Mapping Editor

As shown in the code snippet in the section Building and saving the image 20, you must create an instance of the **SyncManager** object and call the method **open()** to start the Mapping Editor.

On startup of the Mapping Editor, a window is presented. This window has a menu

with an item labeled **Configuration** that allows you to:

1. Start the creation of a new configuration, or
2. open an existing configuration



For the creation of a new configuration, a wizard is used to collect the basic information for the configuration. This is explained in the section Creating configurations 43.

4.2.2 Preferences menu

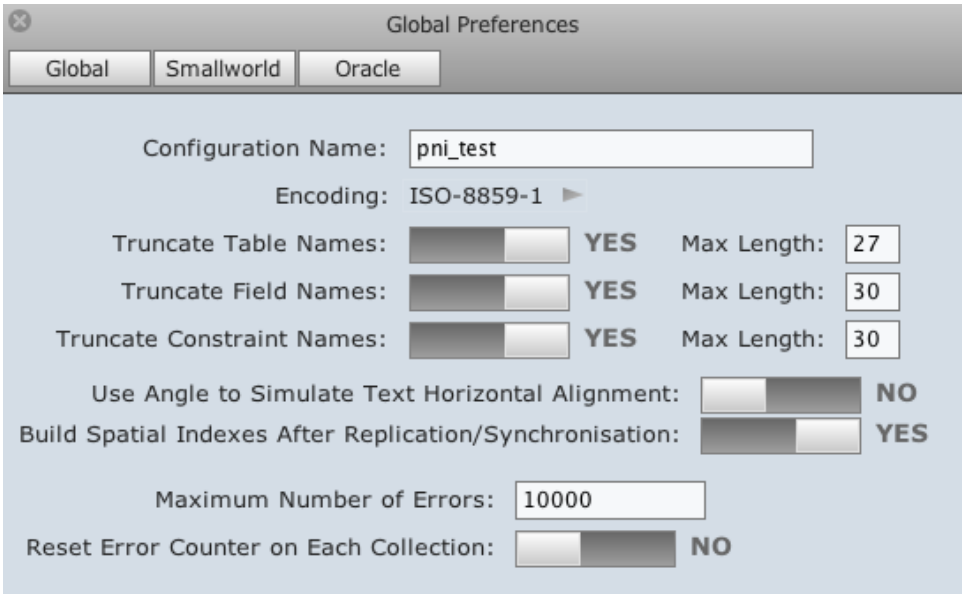
The Preferences menu will be shown when you select **New Configuration** from the **Configuration** menu and it can also be started by selecting **Preferences** from the Contextual Menu (right-click menu) 31.

The Preferences menu contains the basic settings of a configuration. Most of these you will have entered when you created the configuration. The information is divided in four sections:

Global Global preferences 27 (as shown by default, as in the picture below)

Small world Preferences for the Smallworld environment 29

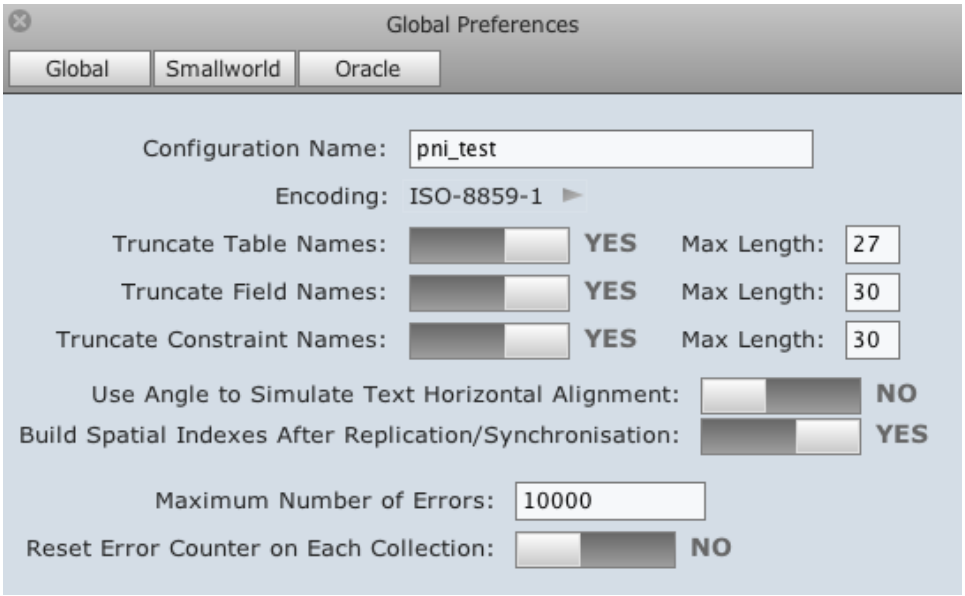
Oracle Preferences for the Oracle database 30



Preferences menu

4.2.2.1 Global preferences

The global preferences tab contains the name of the configuration and several options of a configuration. The options are explained below.



Global preferences window

Configuration Name

This name will be used for the DS alternative created in Smallworld at the time of the synchronization as well as for the .xml file that contains the mappings. It can have a maximum length of 32 characters. The configuration name can not be changed for a

configuration that has been replicated unless destroy datamodel is invoked.

Use Encoding

This option allows you to specify the character encoding used in your Smallworld environment.

Truncate Table/Field/Constraint names and maximum length

Names in Smallworld may be too long to be valid in the Oracle environment and would cause the data model to fail. These three options, when selected, directs *SyncManager* to solve any such problems by truncating and modifying the truncated names with a random number to ensure uniqueness. If left unchecked, any names that are too long are marked with a red mark on the mappings page. You should then modify these names by hand.

Note: These options only truncate names when tables are added (on the mappings page). Once a table is mapped, changing these options has no effect.

Use Angle to Stimulate Text Horizontal Alignment

Oracle offers no equivalent for a horizontal alignment of labels on the left side of a point. In Smallworld this is an attribute of the label style. When you select this option, a 180 degree rotation is added to the alignment of labels. This causes labels that by default would be to the right of a point in Oracle, to rotate to the left side of the point. It is left to the application to flip the labels that thus will be upside-down, back to a legible position.

Build Spatial Indexes After Replication/Synchronization

When this option is selected, SyncManager will create spatial indexes after synchronization / replication. Alternatively these indexes can be created with a command line utility. See the Appendix 56.

Maximum Number of Errors

This parameter allows you to limit the number of errors during synchronization to the value supplied. The counted errors include all errors that appear on the Smallworld or Oracle part of the synchronization process. All errors are logged in a log file, and Oracle Insertion errors are also logged in a table in Metadata.

Reset Error Counter on Each Collection

When this option is activated, the **Maximum Number of Errors** is enforced separately for each collection that is synchronized, otherwise it is a global parameter. For example, if **Maximum Number of Errors** is set to 1000 and 2 collections are to be processed we have the following behaviours depending on the status of this option:

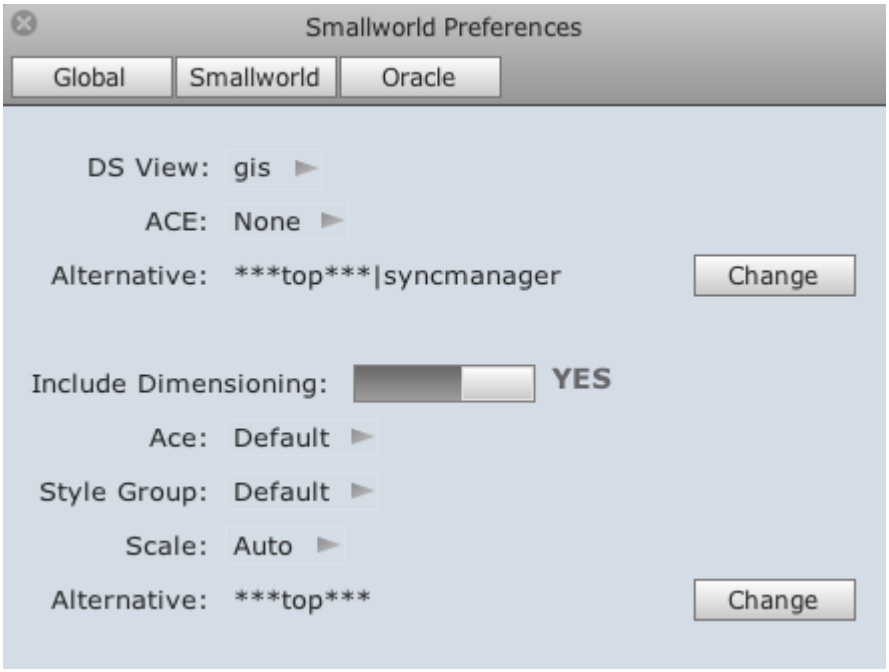
- **Yes:** If the first collection yields less than 1000 errors, the error counter is reset after the collection has been processed and the next collection is also allowed to have a maximum of 1000 errors.

- **No:** If the first collection yields less than 1000 errors, at the start of the processing of the second collection the error counter starts at a value equal to the number of errors encountered with the first collection (x), therefore the second collection may only have a maximum of 1000 - x errors.

4.2.2.2 Smallworld preferences

This tab in the preferences menu contains information about the Smallworld environment.

The information is entered in the wizard for the creation of the configuration. This section is *read-only and is provided for reference only*. If the information in this section is wrong, your only option is to create a new configuration with the correct parameters.



Smallworld preferences window

DS View

Internal name of the Smallworld database view.

ACE

The name of an ACE. Selecting **None** as the ACE to use, means that all the collections of the selected View will be available. Otherwise only the collections in the selected ACE will be available.

DS Alternative

The full pathname of the Smallworld alternative that must be used in the synchronization. You can not select the top (root) alternative. Typically you will use a child alternative that is either specifically created for *SyncManager*, or an existing alternative that fits the role. See also chapter Default merge behavior 18 of the installation manual on merge behavior.

Include Dimensioning?

If you select this option, objects of the type **dimension_holder** will be available for mapping. If you want to include dimensioning, set the **Ace**, **Style Group**, **Display Scale**, and **Style Alternative** to define the Smallworld style in which to export the dimensioning information.

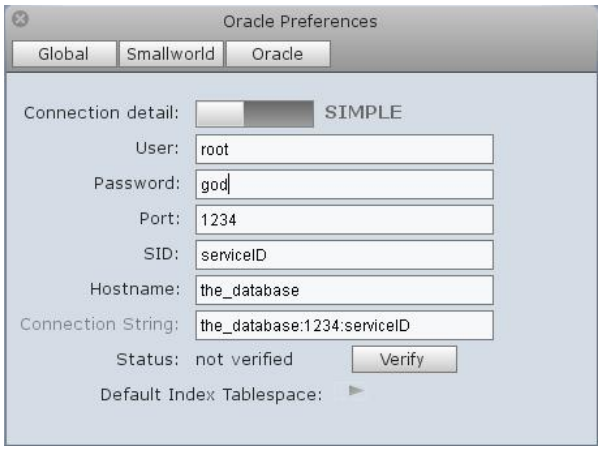
The **Change** button recalls the dialog window where you can select the alternative for which the synchronization will be done. See also chapter Creating configurations 43.

4.2.2.3 Oracle preferences

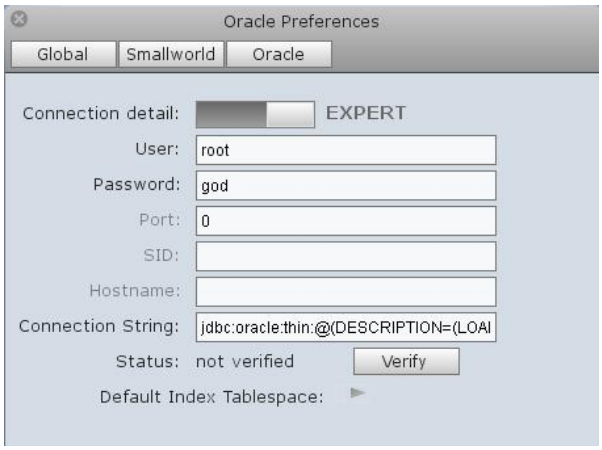
These preferences apply to the Oracle database to be used in the synchronization.

This information was initially gathered when you created the configuration. The validity of these parameters is automatically checked (provided of course the targeted Oracle database is up and running). See also chapter Creating configurations 43.

If these parameters aren't valid the text-box will turn red. You may modify or complete this information later on the configuration details page of the definition window.



Oracle preferences Simple mode



Oracle Preferences window - Expert mode

Connection detail

Simple or Expert mode. In simple mode, you fill in the values in the fields port, sid and hostname, after which SyncManager will build the connection string. In Expert mode, you can fill in the connection string yourself when you need more advanced parameters. In expert mode the other fields are read only.

User Oracle username

Password Oracle password

Port Port to access Oracle database

SID Oracle System ID

Hostname The host of the Oracle database

Status/Verify

The **Verify** button checks if the credentials provided above are valid for a connection with the database. The **Status** can be *not verified* (if the credentials have not been checked), *verified* (if the credentials have checked out) or *failed* (if the credentials are refused).

Default Index/Spatial Index Tablespace

These options set the default tablespaces for indexes and spatial indexes.

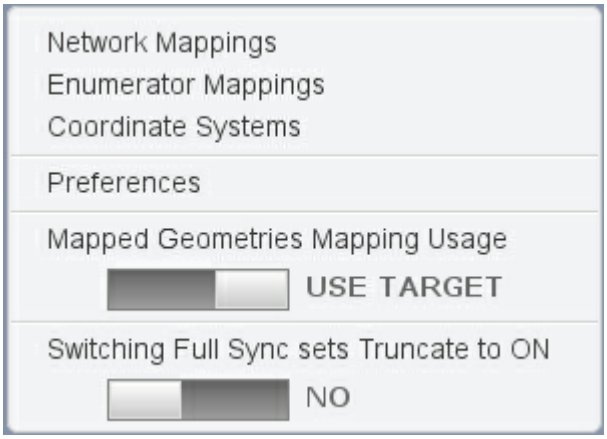
Note: You need a separate user / schema in Oracle for each configuration.

4.2.3 Contextual Menu

After starting the application and creating or opening a configuration, you can access details of how to map the Smallworld data to the Oracle environment through the Contextual Menu. This menu is shown when you right click in the work area of the SyncManager window.

The Contextual Menu contains the following items:

Network Mappings	Mapping of networks 33
Enumerator Mappings	Definition of mappings between Smallworld enumerators and Oracle tables 32
Coordinate systems	Mapping of coordinate systems between Oracle and Smallworld 34
Preferences	Preferences menu 26



Contextual Menu

Mapped Geometry Mapping Usage

This option can be set to use **Sources** or **Target** to control which geometries are available in the list of geometries for mapping.

Set **Target** to make available for mapping only the mapped geometries. Set the switch on **Sources** to make available for mapping only the source geometries.

For example the *Route column*, in the *target* data source it's a single column named *Route*, while in the *source* data source there will be three columns named *Route Inactive*, *Route Active* and *Route Abandoned*.
If you set the switch on **Target**, the columns *Route Inactive*, *Route Active* and *Route Abandoned* will be available for mapping. If you set the switch on **Sources**, only the column *Route* will be available for mapping.

Note: There can't be both target geometries and source geometries mapped at the same type. If you map a target geometry, the switch on Sources and map a source geometry, the previously mapped target geometry will be replaced by the source geometry.

Switching Full Sync sets Truncate to ON

This option, together with the **Full Sync** parameter in the Table Info Page 37 of a certain table determines how the synchronization process will proceed.
The possibilities with **Full Sync sets Truncate** to YES or NO are:

	Full Sync: YES	**Full Sync: NO**
YES	Use only INSERTs for all entries in this table.	Only INSERT new records.
NO	Use only UPDATEs for all entries in this table.	Only INSERT new records.

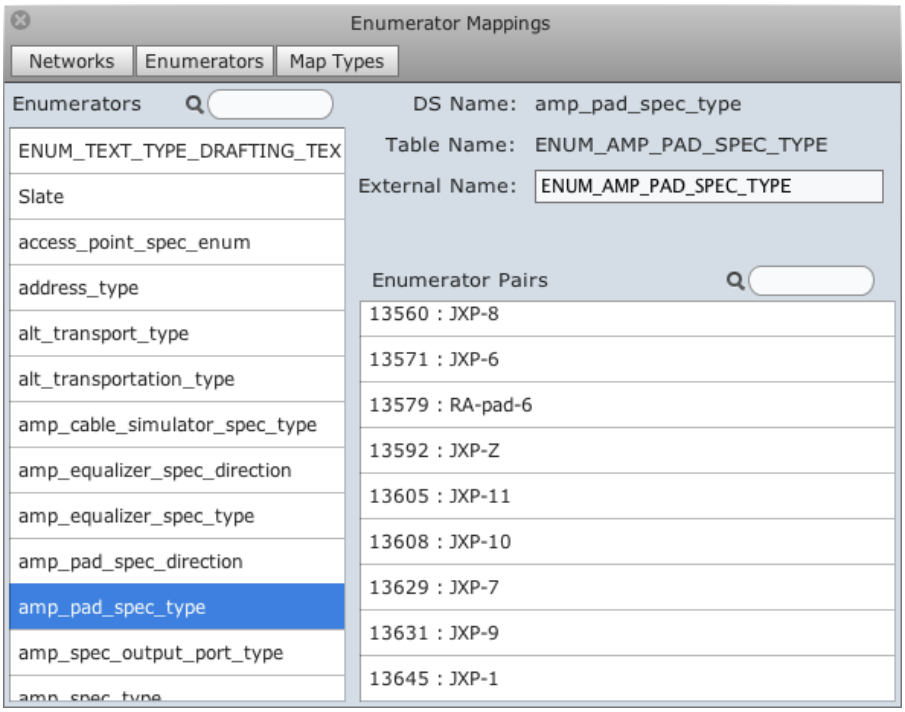
The option "**Switching Full Sync sets Truncate ON**" is by default turned off. This means that when switching the "**Full Sync**" to yes for a collection, next synchronization the collection will be refreshed - ie all records will be sent as updates and missing records will be intercepted and treated as inserts. If the "**Switching Full Sync sets Truncate ON**" is YES, a full synced collection will be replicated - ie all records will be sent as inserts. Some customers do a re-export on a collection when there are differences in Oracle table and smallworld collection for any unexpected reasons or when there is a new field in a mapping they want to initialize.

The best solution (truncate or not truncate table) is to set the **TRUNCATE** switch to YES. If that is the case, the table in Oracle will be truncated when the configuration is saved and all records will be processed as new records. Adding a new field to a mapping does not require a full export on the collection unless the field is new in Smallworld as well (i.e. it appears in the Data Dictionary Updates window.) When adding a new field mapping, the field will automatically be populated, next synchronization, for all records in Oracle. In this version it is not possible in the GUI to set "**Full Sync**" status on field mappings, but this treatment is applied on new field mappings automatically. In one of the next versions, a "**FULL Sync**" switch will also be available on the column mappings as well.

4.2.3.1 Enumerator Mapping page

Both joins and enumerators are treated differently in Oracle. The *Enumerator Mapping page* can be accessed through the Contextual Menu 31 and allows you to view and modify the supporting tables that need to be created to consolidate these differences.

Enumerators, implemented as a field type in Smallworld, have to be translated into separate tables, using a foreign key relation in Oracle. The Enumerators page shows all the tables that will be created to map the enumerator fields found in the collections to be mapped from the Smallworld environment. The page also displays the enumerator pairs, the DS name and the name of the table to be created in Oracle.



Enumerator Mapping page

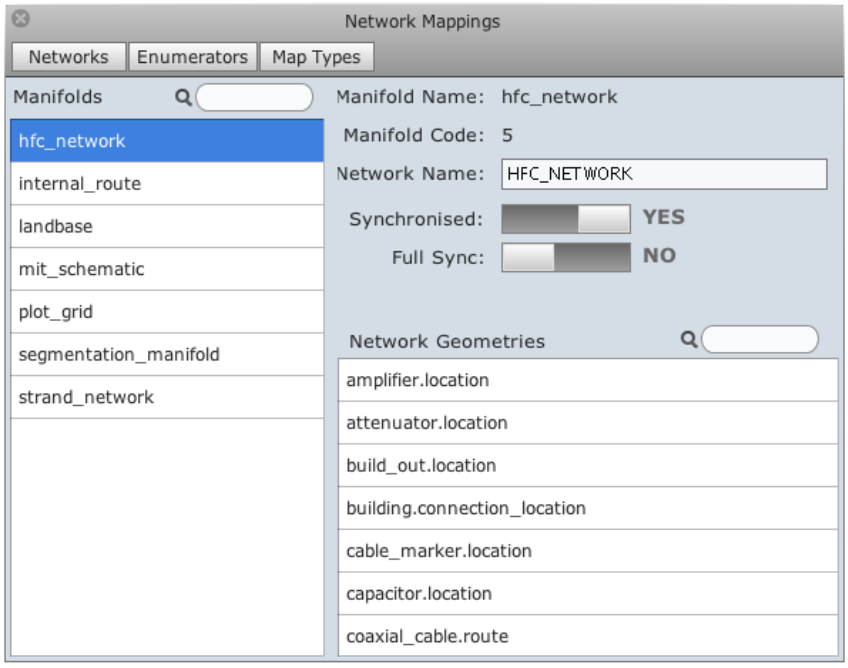
When you select any name in either list, the details of the mapping, including the (Oracle) table name or names are displayed on the right-hand side off the page and can be modified if needed. Where relevant both the (internal) name of the table, and the external name as available to the application, can be modified.

Buttons **Networks** and **Map Types** change the window to Network Mapping Page 33 and Map Types Page 34, respectively.

4.2.3.2 Network Mapping page

The networks used in Smallworld can be mapped to networks in the Oracle environment. Networks are created from Smallworld manifolds.

The *Network Mapping page*, which can be started from the Contextual Menu 31 allows you to define these mappings.



Network Mapping page

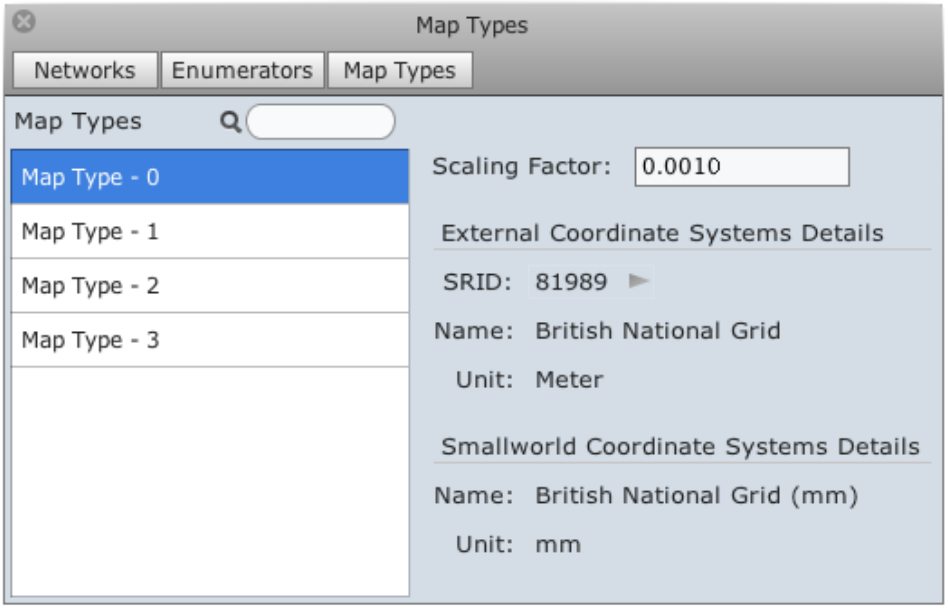
When you select a network from the list of networks in Smallworld (Manifolds), the details of its mapping are displayed on the right hand side of the page. These details include the *Manifold Name, Manifold Code,* the *Network Name*, wether the network is synchronized or not, and a list of geometries included in the network.

The buttons **Enumerators** and **Map Types** change the window to Enumerator Mapping Page 33 and Map Types Page 34, respectively.

Note: A network can be full exported from Smallworld to Oracle only when full synchronization is applied to the geometries contained in the network.

4.2.3.3 Map Types page

The Coordinate systems used in Smallworld have to be mapped to the available coordinate systems in Oracle. The *Map Types page,* which can be opened from the Contextual Menu 31 allows you to define these mappings.



Coordinate Systems menu

When you select a coordinate system from the list of coordinate systems in Smallworld (Map Types), the details of its mapping are displayed on the right hand side of the page. These details include a scaling factor and the ID and name of the coordinate system to use in Oracle.

When a Map type is selected with a double click, the dialog will close.

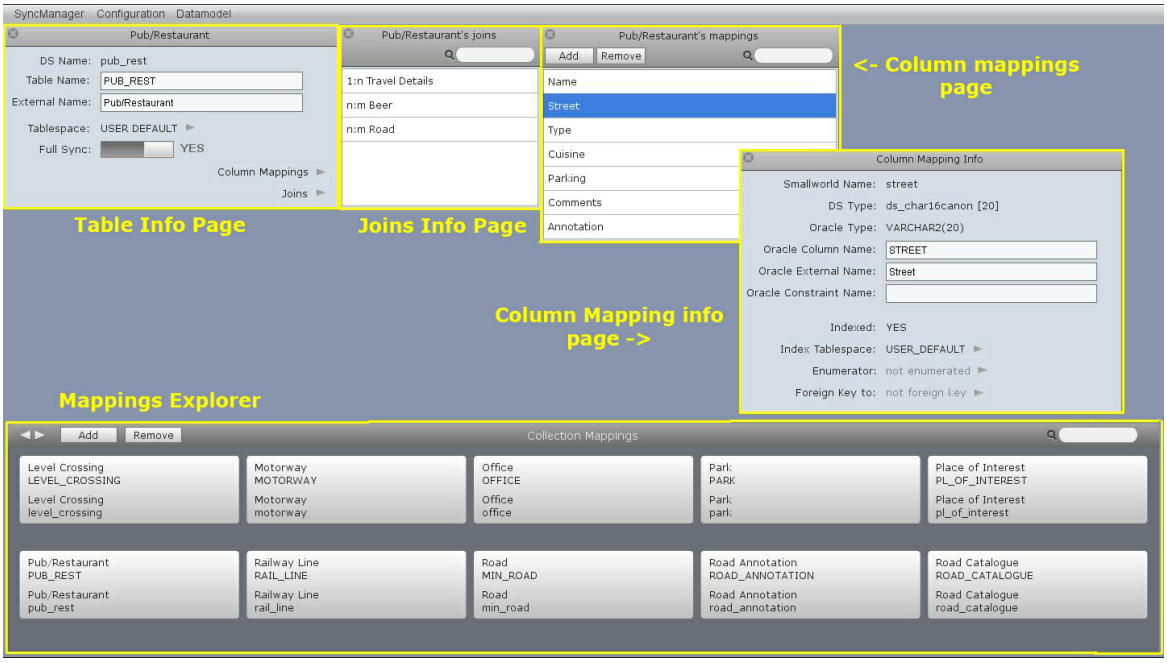
Note: Any non-earth coordinate system (i.e. Cartesian map coordinates) for which no mapping is defined, will be mapped (scaled) to millimeters in Oracle.

4.2.4 Mappings page

The Mappings Page allows you to specify the collections that are to be mapped to the external data source (the Oracle database). You may also override any names generated for these mappings, providing your own names for these mappings. The name of the Table can be modified in the Table Info Page 37, and the names of the columns can be edited in the Column Mapping Info Page 40.

The mappings are created by selecting tables from the Mappings explorer 36 at the bottom of the window.

When a table is selected, *SyncManager* will open the Table Info Page 37 with general information on this table. From this page you can access the Column Mappings page 38 and Joins Info page 42.



Mappings page

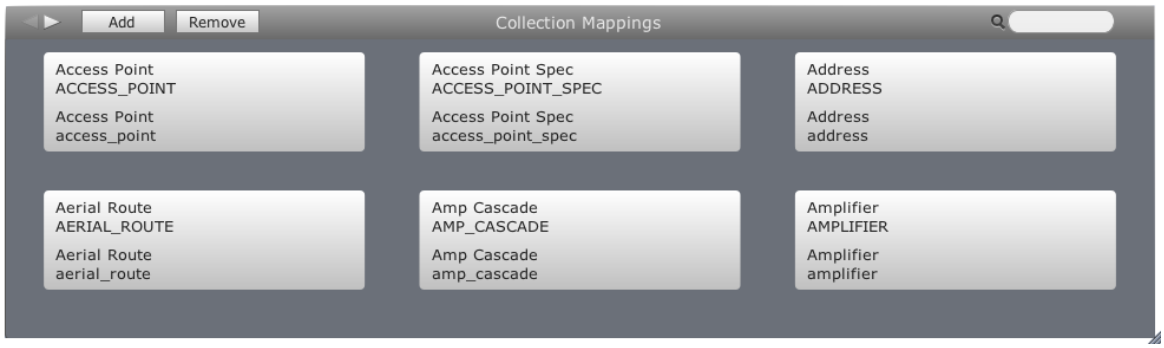
4.2.4.1 Mappings Explorer

The *Mappings Explorer*, displayed at the bottom of the Mapping Editor 25 lists all mapped tables.

The *Mappings Explorer* opens after you select a DS View in the Smallworld Preferences 29 dialog.

If you double-click on one of the mapped tables, the mapping details of the specific table are displayed in the Table Info page 37. Also displayed are the related menus with the Column Mappings 40 and Joins 42.

A single-click will only show the Table Info page 37.



Mappings explorer

4.2.4.2 Table Info page

This page contains information how a table, identified by **DS Name**, should be synchronized. This page is shown when you click a table in the Mappings Explorer 36.

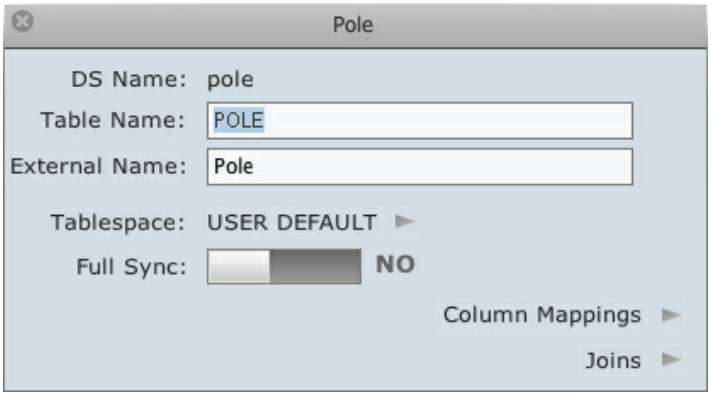


Table Info page

DS Name

This identifies the name of the collection for which the info is provided.

Table Name

The name of the Oracle table that will receive the data.

External Name

The external name of the Oracle table that will receive the data.

Tablespace

The tablespace used to map this collection.

Full Sync

Synchronization mode: *Full* or *Partial*.

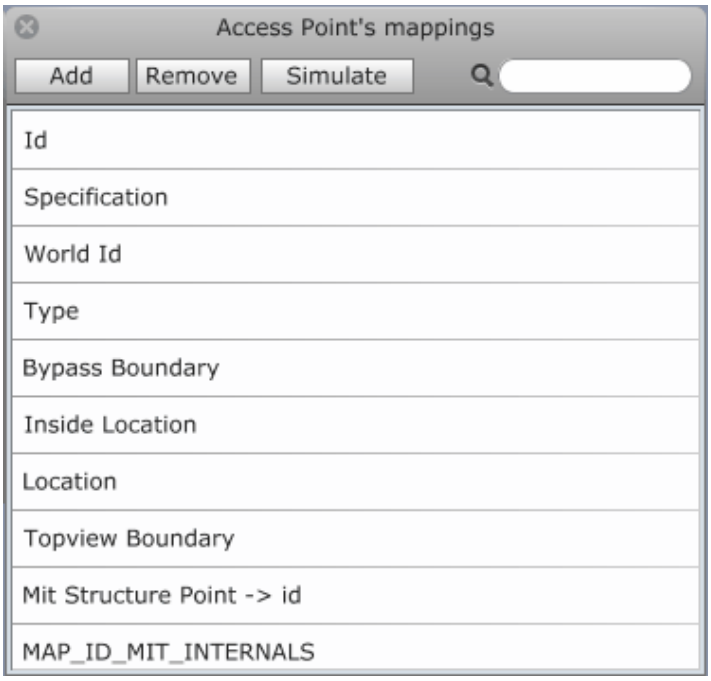
At the initial replication for a given configuration, **Full Sync** will be set to **Yes.** At subsequent synchronizations, **Full Sync** parameter together with the **Switching Full Sync sets Truncate to ON** option from the Contextual Menu 31 determine how the synchronization will proceed.

How is described in the table given in the section Contextual Menu 31.

The column mappings and joins are displayed for a mapped table if you click resp. on the **Column Mappings** 38 button or the **Joins** 42 button, so you can view the field mappings or the joins of the table.

4.2.4.3 Column Mappings page

The Columns Mappings page, which is part of the Mappings page 35, contains the specification of the fields that should be synchronized.



Column Mappings page

You can add or remove column mappings for this table by using the **Add** and **Remove** buttons.

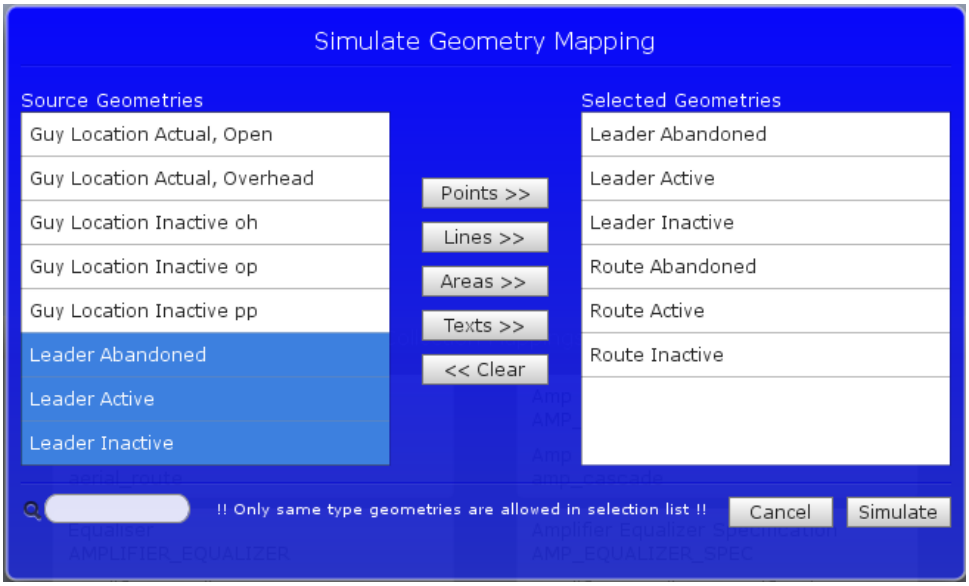
By clicking on one of the fields in the list, the Column Mapping Info page 40 opens. Here you can see and edit [??] the properties of the field.

Geometry mappings

If one of the columns in the table has more then one geometry of the same type (i.e. the geometry mapping is homogeneous), you can use the **Simulate** button to simulate the geometry mappings. This will open the Simulate Geometry Mapping dialog 38.

4.2.4.4 Simulate Geometry Mapping dialog

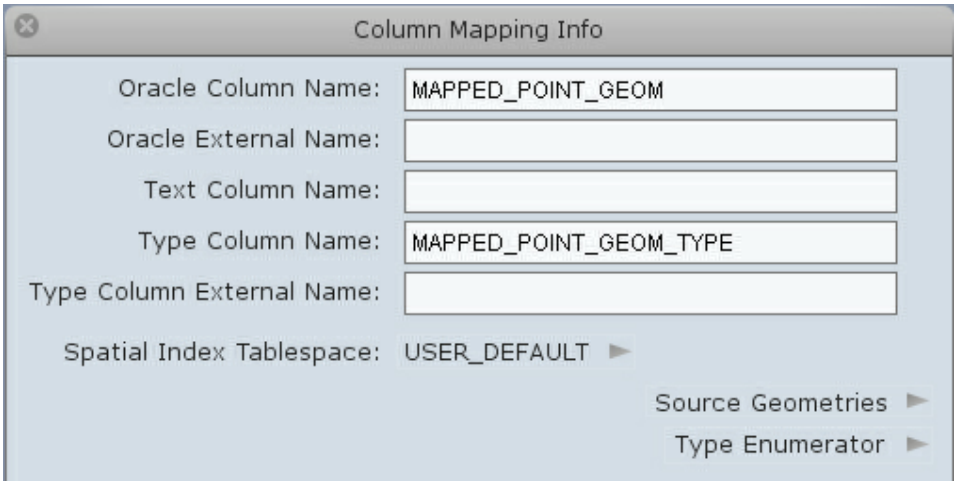
When additional information regarding a geometry mapping is required or there are too many geometries, of same type, that can be grouped into one column then the simulate geometry mapping can be used. The "Simulate" button is only active on the Column Mappings Page 38 when there are enough source geometries that can be grouped.



Simulate Geometry Mapping Dialog

Use the **Add** button to move the geometries from the list of **Source Geometries** in the left to the **Selected Geometries** list to the right. When you're done click on the **Simulate** button to simulate the geometry mapping. This will group the geometries in the Selected Geometries list to the right into a single geometry of the same type and open a **Column Mapping Info** page for this geometry.

In addition to required geometry column(s), the simulate action will create an enumerator based on the source geometry names and a new column that will hold the new enumerated value.



Column Mapping Info Page - Simulated Geometry

The resulted **Column Mapping Info** page contains the following items:

Oracle Column Name The name for the Oracle column that will contain the Simulated Geometry.

Oracle External Name The external name for the Oracle column.

Text Column Name In case the Simulated Geometry is a Text Geometry, the actual text will be stored in this column.

Type Column Name The name of the column that stores the IDs for the types of the source geometries, as foreign keys to the Type Enumerator table.

Type Column External Name The external name of the above column.

Spatial Index Tablespace The tablespace to be used for the spatial index.

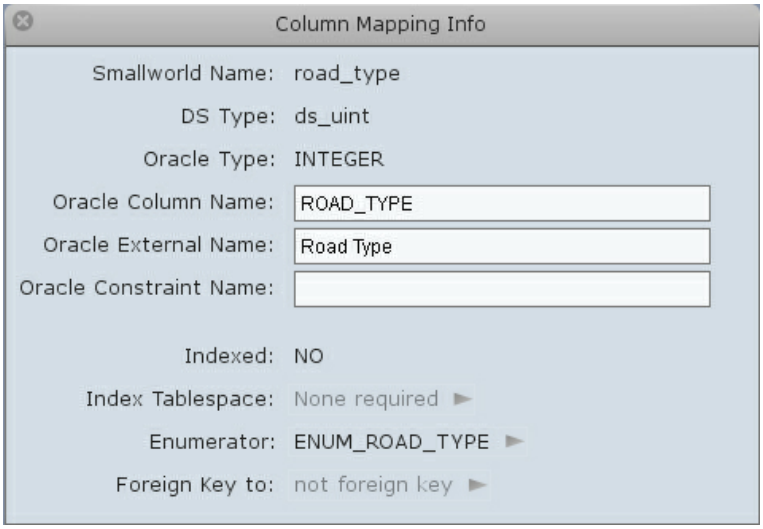
Source Geometries Clicking this button will show a list of the geometries selected to create this Simulated Geometry.

Type Enumerator Clicking this button will open the Enumerator Mapping Page 32 for the enumerator created for the geometry types included in this simulation.

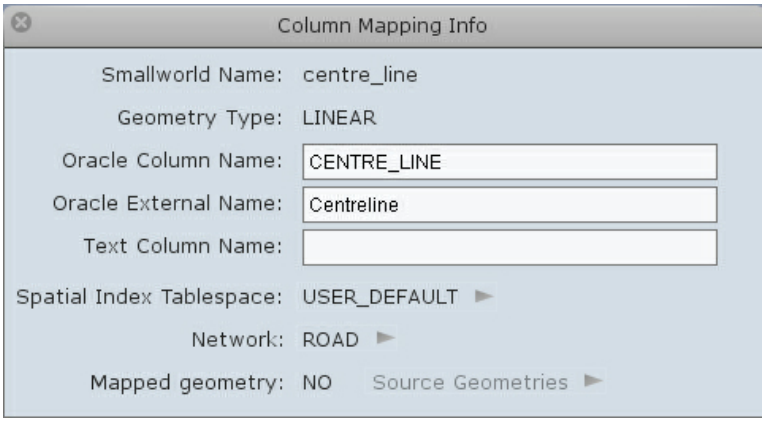
Note: Only geometries of the same type are allowed in the Selected Geometries list.

4.2.4.5 Column Mapping info page

The *Column Mapping Info page* contains the details of a field/column mapping. The dialog is opened by selecting a field in the Column Mappings page 38.



Column Mapping Info - enumerated



Column Mapping Info - geometry

The following properties of a field/column are displayed:

Smallworld Name

DS type

Oracle type

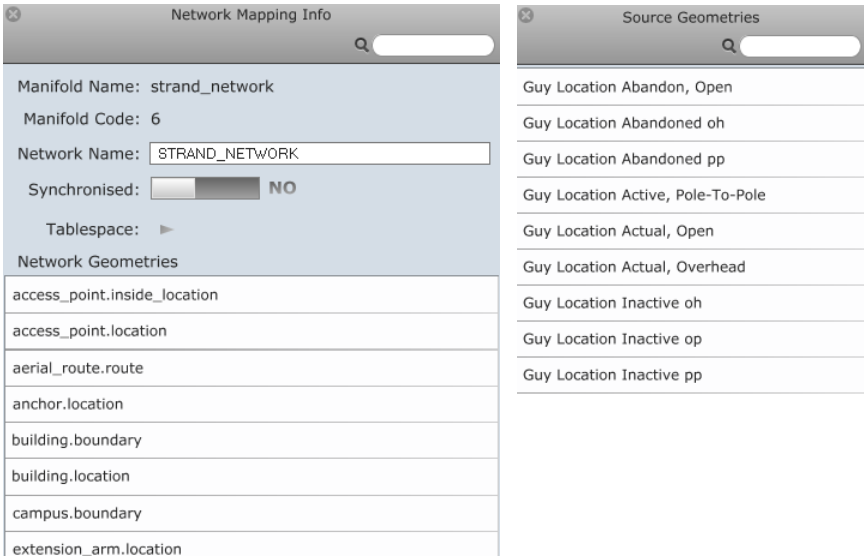
Oracle Column Name

Oracle External Name

Oracle Constraint Name

Depending on the type of the field/column, other information will be available.

- If the field is **enumerated**, you can click on the Enumerator button ("ENUM_ROAD_TYPE >" in the example) to see Enumerator Mapping Information 32.
- If the selected field is a **geometrical field**, the Column Mapping Info page shows the Network and the Source Geometries for the selected field. Click on the button **Network** ("ROAD >", in the example) to view Network details like Manifold Name and Code, Network Name and also the Network geometries that the network supports. Click on the button **Source Geometries >** to view the possible source geometries of the selected field.



- If the selected field is a foreign key to another collection, this will be shown on the column's mapping info page. The joins for a table can be shown by the button **Joins** in the Table Info page 37. This will display the Join Info page 42.

4.2.4.6 Join Info page

The *Join Info page* displays the joins of a selected table. The dialog can be opened from the the Table Info page 37.

Join Info Page

Defining many-to-many relations in Smallworld have to be mapped using an intermediate table in Oracle. The intermediate tables that will be created in the Oracle data model to support the join relations found in the mapped collections from the Smallworld environment are listed at the end of the collections list (for example **int!mit_rme_port_copper_crv**).

Click on a join in the *Join Info page* and the Table Info Page 37 will be populated with 1) either

the collection with which the specific table is joined, or 2) the intermediate table that will be created.

4.3 Mapping Configurations

SyncManager does the synchronization between Smallworld and Oracle on the bases of configurations created by the Mapping Editor 25. These *Mapping configurations* are the building block of the *Syncmanager*.

Multiple configurations may coexist, thus allowing you to synchronize different aspects of the information contained in your Smallworld application.

Mapping configurations are stored in an XML-file for every configuration. Where these files are stored is explained in the section Environment settings 18 of the *Installation Guide*.

This chapter explains step by step the procedures to create and modify mapping configurations using the tools provided by the Mapping Editor 25.

The final section Updating configurations 46 explains how to deal with changes in the Smallworld data model, and how to generate and maintain the corresponding data model in Oracle.

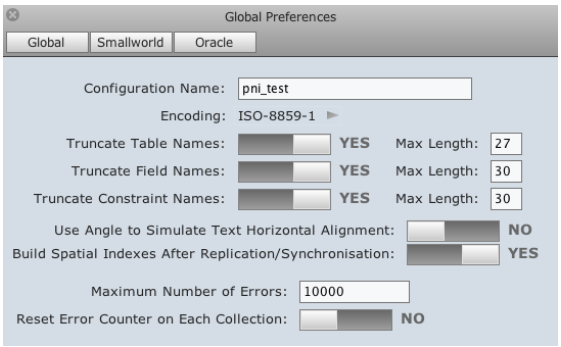
4.3.1 Creating configurations

Using the commands from the Configuration entry in the menubar of the Mapping Editor 25, you can create, open and save configurations.

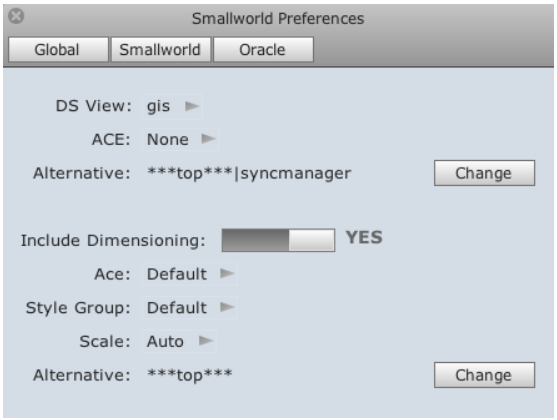
When creating a new configuration, a wizard is invoked to collect the basic information.

Step 1: Select the option **Create** in the configuration menu. A wizard will be started to guide you to the process.

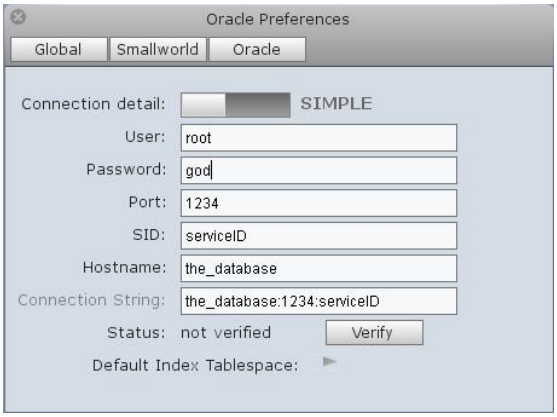
Step 2: Enter the fields for the Global preferences 27. You may modify or complete this information later in the same section of the Preferences menu 26.



Step 3: Fill in the parameters for Smallworld 29: The Database view, Ace, Alternative and Dimensioning Information. Note that this information cannot be changed afterwards.



Step 4: Enter the connection parameters of the Oracle database. If these parameters aren't valid the text-box will turn red. You may modify or complete this information later in the Oracle preferences 30 of the Preferences menu 26.



Step 5:

A configuration, like any document, can be saved and / or closed when you have finished working on it. You will find the commands for this on the **Configurations** menu.

You can delete a configuration by deleting the xml file (stored in the mapping file repository, see chapter Environment settings 18) that defines it . Note that this does not delete a data model created in Oracle based on the configuration, any synchronized data, or any metadata created in the data store.

4.3.2 Defining Mappings

This section contains a step by step description on how to create mappings. To start this you must first set the Smallworld Preferences 29.

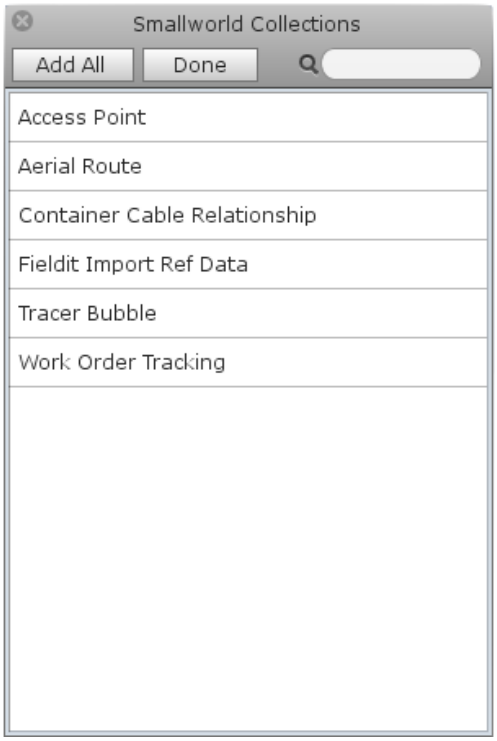
Step 1:

First, click **Add** in the Mappings Explorer 36. To bring up the **Smallworld Collections Dialog**. There you will find a list of all the collections available in Smallworld for the DS View and ACE you selected in Smallworld Preferences 29. In this dialog you can select the collections you

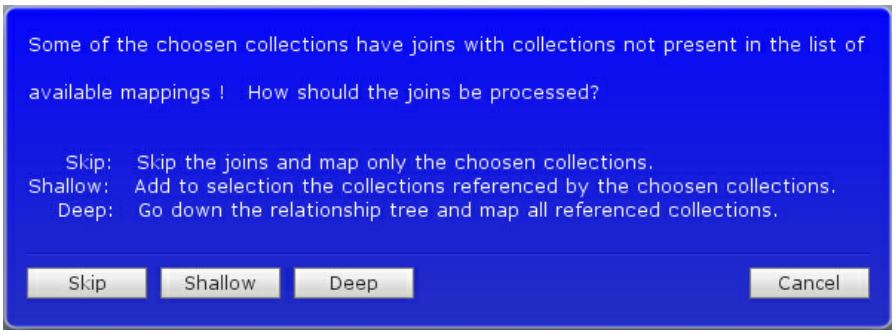
would like to synchronize or click **Add All** if you want the whole dataset. Click **Done** to close the dialog and map the collections. Note that after the collections have been mapped they will not appear in this dialog any more; this dialog contains only available Smallworld collections that have not been mapped.

If any of the collections you selected have join fields, then when you click **Done** in the **Smallworld Collections Dialog** you will be prompted by the **Join Handling Message Box**. Here you can choose to **Skip** the joins, do a **Shallow** processing of the joins (go down the relationship tree only one step), or perform a **Deep** processing of the joins, effectively mapping all the collections that have a relationship path with the collections you have selected.

Collections can be added later also in the same manner. For more information, please see Add a new collection to the Mapping Configuration 47.



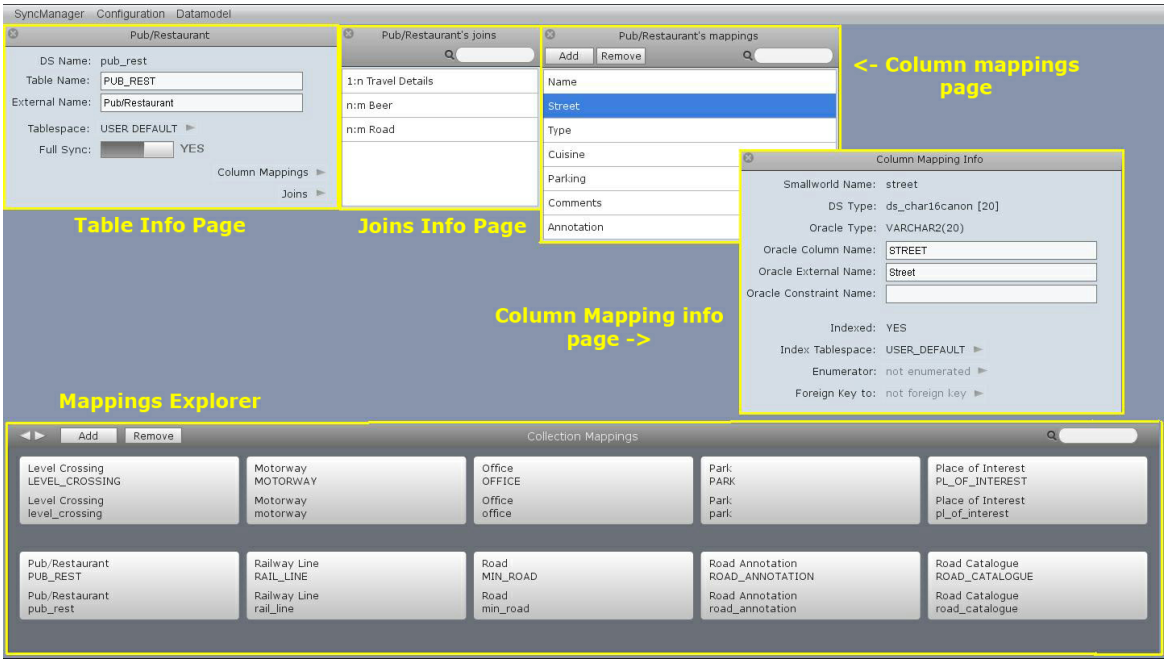
Add Smallworld collections Dialog


Some of the choosen collections have joins with collections not present in the list of available mappings ! How should the joins be processed?
Skip: Skip the joins and map only the choosen collections.
Shallow: Add to selection the collections referenced by the choosen collections.
Deep: Go down the relationship tree and map all referenced collections.
Skip
Shallow
Deep
Cancel

Join Handling Message Box

Step 2:

To view or edit the mappings for a specific collection you have mapped in the previous step, find it in the Mappings Explorer 36 and double-click on it. This will bring up the dialogs of the Mappings Page 35. Here you can alter specific mappings for each collection or field of a collection. For more information on this, please see Mappings Page 35, and its subchapters.



Mappings page

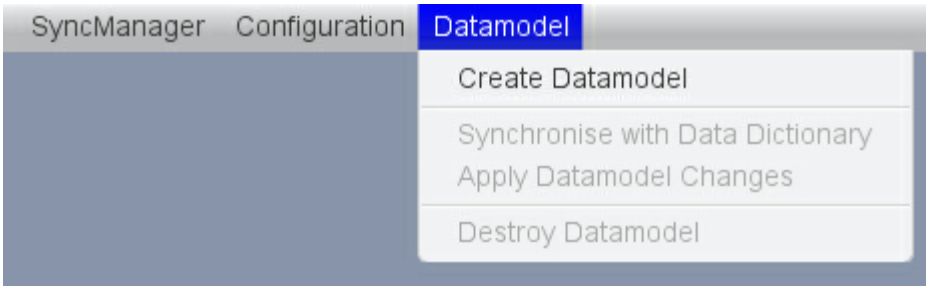
4.3.3 Updating configurations

This chapter completes the information on mapping configurations by explaining how to use *SyncManager* to ensure that your synchronization is based on a valid mapping of the data models in the Smallworld and Oracle environments.

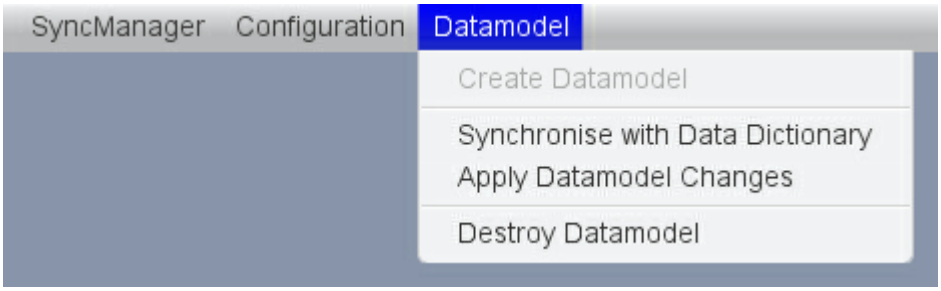
The **Datamodel** entry in the menu bar of the Mapping Editor 25 contains several commands you can use to update your datamodel in Oracle when your Smallworld environment has changed.

Recall that in order to make synchronization possible, your configurations maps (part of) your data model in Smallworld to a data model in the Oracle environment. Also note that the data model in your Oracle database is actually created or updated when synchronizing.

The first time you will create the mapped data model in Oracle. From there on you can update it.
After updating or changing a configuration, you always need to export the mapped data model to your Oracle environment.
The commands to create or update the data model, and to optionally destroy the data model in Oracle, can be found on the menu **Datamodel**:



Datamodel menu (before creating a datamodel)



Datamodel menu (after creating and editing a datamodel)

The commands to create or update the data model, and to optionally destroy the data model in Oracle, can be found on the **Datamodel** menu. These commands are:

Synchronize With Data Dictionary 47
Add a new collection to the Mapping Configuration 47
Add a new field mapping to a collection mapping 49
Remove a field mapping from the Mapping Configuration 49
Remove a collection mapping from the Mapping Configuration 49
Re-export a configuration 50

4.3.3.1 Synchronize with Data Dictionary

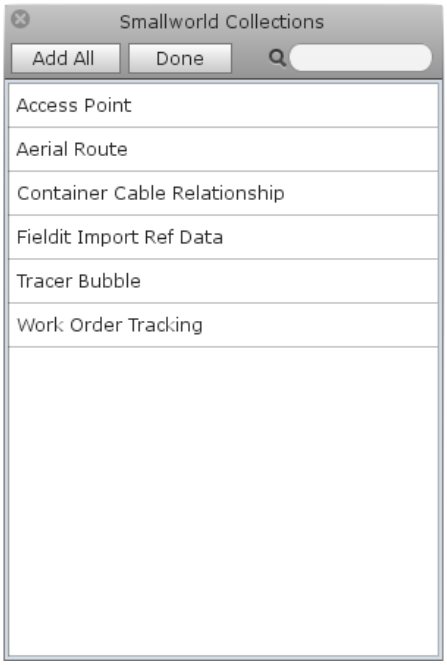
This command, found on the **Datamodel** menu item will look at the data model in your Smallworld environment and, for the selected database view and ACE, it will:

1. Remove obsolete tables / fields from your configuration
2. Remove any changed fields

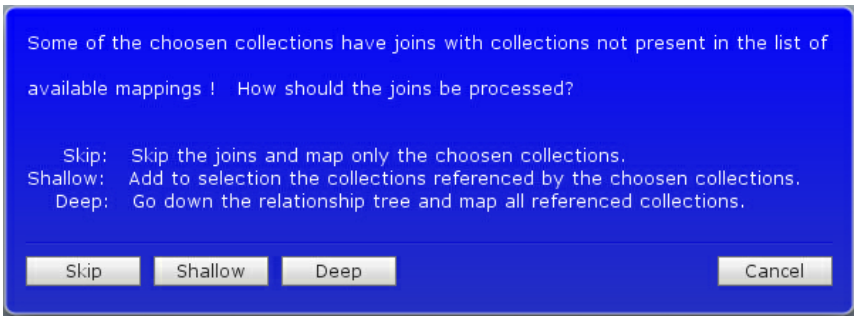
4.3.3.2 Add new collection

You can add a new collection to your Mapping Configuration by following the next steps:

1. On the Mappings explorer 36 you will see a list of the mappings that exist in the current configuration.
2. Click on the **Add** button.
3. > a list of available collections appears (excluding the ones already mapped), select the collections that you would like to add to the mapping configuration or click **Add All** to select all the collections, and click on the **Done** button.



4. If any of the newly added mapping collections has a parent/child relationship with another mapping collection, a dependency dialog will appear. This allows you to automatically add the related mapping collections.


Some of the choosen collections have joins with collections not present in the list of available mappings ! How should the joins be processed?
Skip: Skip the joins and map only the choosen collections.
Shallow: Add to selection the collections referenced by the choosen collections.
Deep: Go down the relationship tree and map all referenced collections.
Skip
Shallow
Deep
Cancel

5. The mapping collection(s) are now added to the list of mappings in the Mappings Explorer 36. Note that these are marked as re-export collections 50.
6. To synchronize the configuration with the database, the datamodel must be updated.
   Go to **Datamodel** menu and use the **Apply Datamodel Changes** command. A warning will appear that tells you that the existing table in Oracle must be emptied manually before continuing. Finally, the datamodel is updated and the configuration is saved.

4.3.3.3 Add new field

You can add a new field mapping to an existing collection mapping by following the next steps:

1. On the Mappings Explorer 36 you can see a list of collection mappings that exist in the current configuration.
2. Double-click the collection mapping you would like to add fields to, and click on the **Add** button in the Column Mappings page 38
3. > a list of unmapped fields will appear from which you can choose to add to the collection mapping. Select the fields you would like to add to the collection mapping or click **Add All** to select all the available fields and click on the **Done** button
4. If any of the newly added fields is a join field to an unmapped collection, the collection will be automatically mapped.
5. To synchronize the configuration with the database you must update the datamodel. Go to **Datamodel** menu and use the **Apply Datamodel Changes** command. A warning may appear saying that the existing table in Oracle must be emptied manually before continuing.
   If the added field does not exist in the Oracle database, a new column is added to the table of the mapping collection. However, this column will not contain any data and only data which will be synchronized will populate the column. If the column should contain the data as is in Smallworld database, choose to re-export the mapped collection before saving the configuration.
   Finally, the datamodel is updated and the configuration is saved.

4.3.3.4 Remove field

You can remove a field mapping from an existing mapped collection in the mapping configuration by following the next steps:

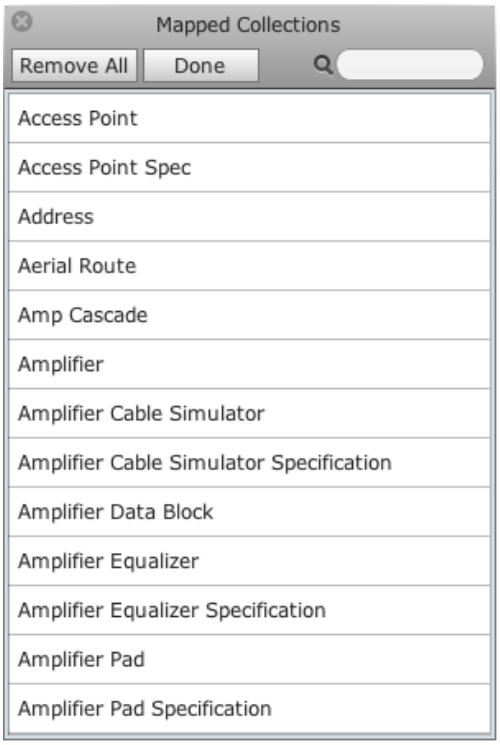
1. On the Mappings Explorer 36 you will see a list of the mappings in the current configuration
2. Double-click the mapped collection you would like to remove fields from
3. Select the fields you would like to remove or click **Remove All** to select all fields and click the **Done** button
4. Save the configuration

Note: In case of removing a field mapping from a collection you do not need to update the datamodel. This is because of the fact that no data is removed by *SyncManager*.

4.3.3.5 Remove collection

You can remove a collection mapping from an existing mapping configuration by doing the following steps:

1. In the Mappings Explorer 36 click on the **Remove** button. The **Mapped Collections** dialog will appear.



2 Select the collections you would like to remove or click **Remove All** to select all the collections, then click **Done**.

3. If the removed collection mapping has relationships with other mapped collections the join fields that those collections have with the removed collection will also be automatically removed.

3. Save the configuration

Note: In case of removing a collection mapping you do not need to update the datamodel. This is because of the fact that no data is removed by *SyncManager.*

4.3.3.6 Re-export configuration

The command **Re-export configuration** from the **Configuration** menu allows you to perform a re-export. Please note that the first time you synchronize, all mapped collections are replicated in full. After this initial replication, only updates are processed. The re-export command allows you to force a complete replication of the marked collection or collections at any time after the initial replication. You can mark a collection for complete replication by activating the **Full Sync** option in the Table Info Page 37 for that collection. The actual replication will take place during the next synchronization. The way the replication will proceed is defined by the use of the **Switching Full Sync sets Truncate to ON** parameter in the Contextual Menu 31. For more information please see the two links in this paragraph.

Once you have updated the configuration, or need to make other changes, you should check, modify, or create the mappings and other information as described in detail in chapter Defining the mappings 31.

Note: You must empty the associated table or tables in Oracle before synchronizing, or the replication will fail.

4.4 Synchronization

Synchronization with *SyncManager* between Smallworld and Oracle is based on Mapping Configurations 43.

This Mapping configuration is used to control the actual transfer of data from the Smallworld environment to the Oracle environment.

This chapter describes how to Initiate the synchronization 51 and how to Analyze and fix possible errors 51.

4.4.1 Start synchronizing

To start synchronizing you need to create an instance of the object **sync_manager** and invoke the method **run()** with the name of the configuration as a parameter. The code snippet below shows a basic example you can enter at the Magik prompt. You may create more elaborate or convenient functionality around or based on this example yourself, but this is beyond the scope of this manual.

See also chapter Building and saving the image 20.

```
_block
     my_sm << sync_manager.new()
     my_sm.run(<<configuration name>>)
_endblock
$
```

The only way to stop the synchronization process before it completes, is to end or kill the java application from the operating system.

4.4.2 Analyzing and fixing errors

At various levels of the synchronization process information is generated and will be recorded in various logs.

Traceback and progress information of the Magik execution may have been redirected to a log file. Dealing with this information is however beyond the scope of this manual. See your Magik documentation for details.

The execution of the synchronization procedure generates information, both general and on the errors encountered. This information is recorded in two separate log-files. The location of these log-files is defined in your environment settings. See chapter Environment settings 18 for details.

The most common error descriptions can be found in the Appendixes 55.

All information and errors (exceptions) related to the transferred data are stored in the *SyncManager* data store. The information can be found in a special database view called **sync_manager** and in an alternative with the name of the Mapping configuration. See appendix syncmanager.ds 57 for a detailed description of the tables in the datastore.

A viewer for this information is currently under development. Until this tool becomes available

and alternatively you can access the information from the Magik prompt. Please use the code snippets below as an example and a guide.

To open the view and access the alternative:

```
smv << sync_manager_view.open()
smv.go_to_alternative(<<configuration name>>)
$
```

For each synchronization attempt a record is stored in the collection **sync_export_info**. A description of this collection is shown below:

```
smv.collection(:sync_export_info).an_element().describe_fields()
$
sync_export_info93:(51529)
*V  id                        51529
 V  sync_mode                 replicate
 V  sync_end_checkpoint       1164118550_SY_CFG_PNI
 V  sync_start_time           21/11/2006 14:15:51
 V  sync_end_time             21/11/2006 14:57:14
 V  sync_records_exchanged    5184474 transmitted - 0 received
    sync_commit_timestamp     1164121033
!VJ sync_exceptions           select_collection(sync_exception)
```

All exceptions (errors) are stored in the collection **sync_exception**. The (join) field **sync_exception** in the **sync_export_info** record provides a link to all exceptions for a given synchronization attempt:

```
smv.collection(:sync_exception).an_element().describe_fields()
$
sync_exception96:(51534)
*V  sync_err_id               51534
 V  sync_err_type             alphanumeric
 V  sync_err_status           offline
    sync_err_change_type      I
    is_solved?                False
    is_smallworld_sourced?    True
    sync_err_source_urn       swrefV…keysZ7122809
    sync_err_table_name       mit_structure_point
    sync_err_field_name       unset
    sync_err_description_id   90178
    sync_err_stack_trace_id   90179
 V  sync_err_sql_code         unset
    sync_export_info_id       51529
 VJ sync_err_description      Unable to transform record for syncronisation
 VJ sync_err_stack_trace
 VJ sync_export_info          sync_export_info93:(51529)
```

There are two types of exception records: alphanumeric and geometry. A geometry exception stores the name of the geometry field that caused the error. This information can then be used to find the geometry involved:

```
exp_rec << smv.collection(:sync_export_info).an_element().sync_exceptions.an_element()
```

```
$
sync_exception96:(51530)

exp_rec.describe_fields()
$
sync_exception96:(51530)
*V  sync_err_id               51530
 V  sync_err_type             geometry
 V  sync_err_status           offline
    sync_err_change_type      U
    is_solved?                False
    is_smallworld_sourced?    True
    sync_err_source_urn       swref…VkeysZ5041402
    sync_err_table_name       riser
    sync_err_field_name       route
    sync_err_description_id   90174
    sync_err_stack_trace_id   0
 V  sync_err_sql_code         unset
    sync_export_info_id       51529
 VJ sync_err_description      Could not extract geometry owning world.
                                [!--- Record Description ---!]
                                Table name : RISER
                                Class name : sk_gis_riser
                                Key Fields :
                                id:  5041402
                                Geometry Field:  route

 VJ sync_err_stack_trace      unset
 VJ sync_export_info          sync_export_info93:(51529)

urn_manager.get_object_from_urn(exp_rec.sync_err_source_urn)
$
riser894:(5041402)

urn_manager.get_object_from_urn(exp_rec.sync_err_source_urn).route
$
simple_chain:(gis_id(2147804718,427214750,5041403))

urn_manager.get_object_from_urn(exp_rec.sync_err_source_urn).route.world
$
unset
```

Part V

5 Appendixes

This section contains additional information on the *SyncManager* product.

1. Commandline utility for indexing 56
2. SyncManager_metadata 57
3. Error descriptions 62
4. Known limitations 64

5.1 Command line utility for indexes

It is possible to have spatial indexes created automatically. This feature is enabled by selecting the option **Build spatial indexes after replication/synchronization** in the configuration details page (see Configuration details page 26).

Alternatively these indexes can be created manually with a command line utility that is delivered with *SyncManager*. This utility can be found in the following folder:

<syncmanagerInstallDir>\roos_sync_manager\modules\src\sync_manager_src\resources\base\data\java\plugins

Usage and options:

Usage:

```
java -jar SyncUtil_1.0.0.jar [options]
```

Options:

```
-host       Address of the Oracle database server.
-port       The listening port. [1521]
-sid        The SID of the Oracle database.
-user       User owning the data.
-password   Password for the user.
-v          Verbose output.
index       Create spatial indexes, metadata, and views.
clean       Delete spatial indexes, metadata, and views.
bounds      Calculate and set the used bounds for every map.
```

The bounds option will calculate the actual used bounds of the map geometry, which often differ from and are more useful than the bounds of the world coordinate system. The used bounds are stored in the table **sync_map** in the SyncManager meta database. See the Realworld Syncmanager metadata 57 description in the Appendix).

5.2 SyncManager metadata

SyncManager stores information regarding exports and mappings in several tables.

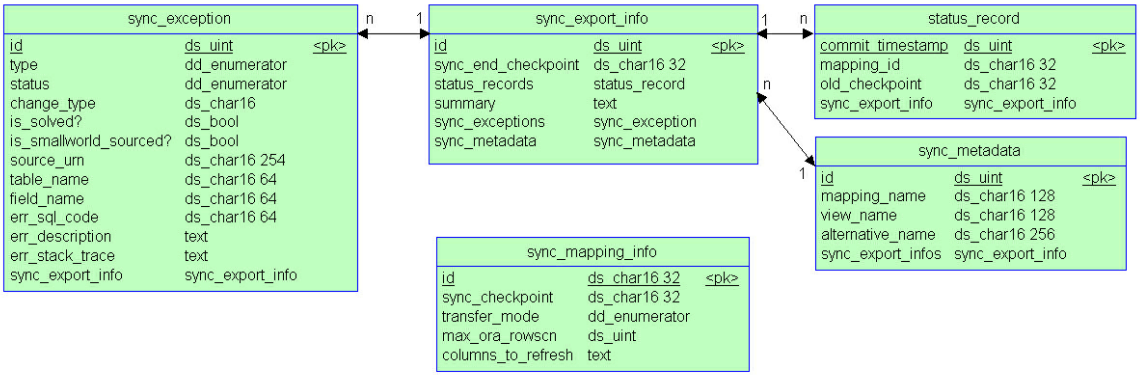
The information that regards the Smallworld side of the mapping and transfer is stored in the *SyncManager* Smallworld dataset **syncmanager.ds** 57.

The rest of the information, concerning Oracle data modeling and mapping is stored in a set of Oracle tables 59.

5.2.1 syncmanager.ds

The main purpose of the Smallworld SyncManager dataset is to provide a history of the exports for internal use (managing checkpoints, choosing transfer modes, restoring data) as well as providing a view for external use (debugging, testing, functionality extending).

Below you find a diagram of the tables in this dataset and a table providing descriptions for the tables and their fields.



SYNC_EXCEPTION	Contains exceptions generated during runs and marks their resolutions.	
	ID	Primary key.
	TYPE	Type of exception.
	STATUS	The status of the exception.
	IS_SOLVED?	Marks if the exception is solved.
	IS_SMALLWORLD_SOURCED?	Marks if the exceptions is smallworld sourced.
	SOURCE_URN	The Uniform Resource Name of the exception source.
	TABLE_NAME	The name of the table linked to the exception.
	FIELD_NAME	The name of the field linked with the exception.
	ERR_SQL_CODE	The SQL code for the error.
	ERR_DESCRIPTION	The description of the error.
	ERR_STACK_TRACE	The stack trace of the error.

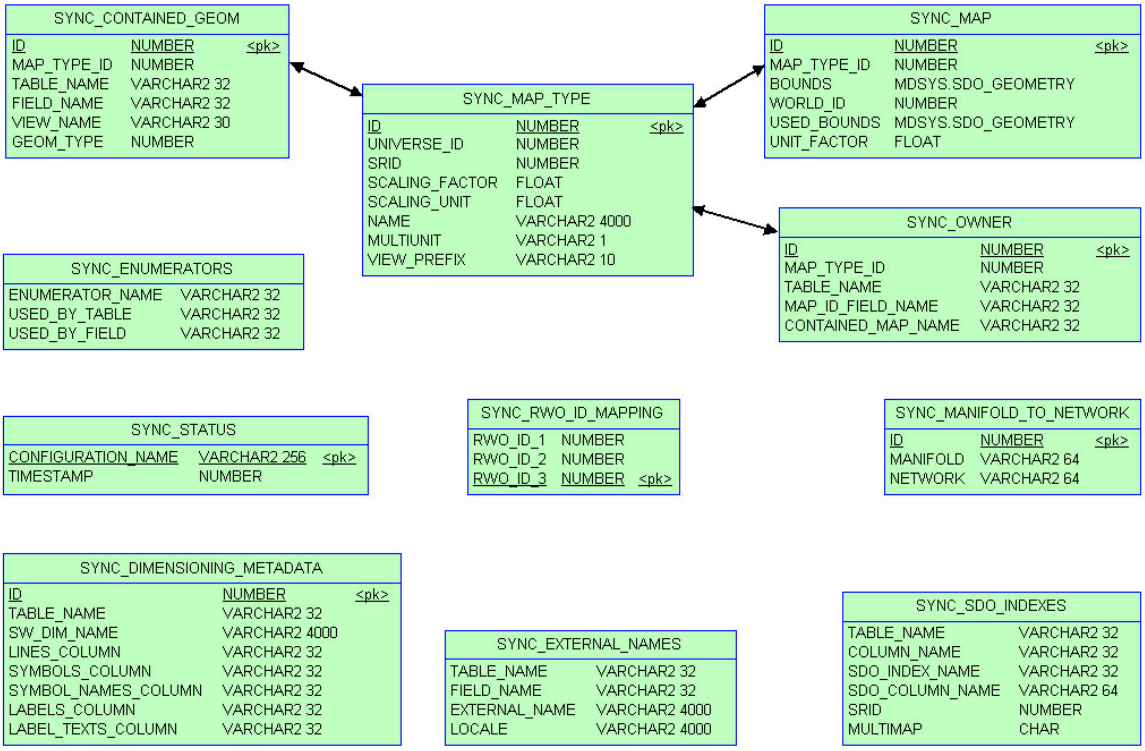
	SYNC_EXPORT_INFO	n:1 join field with the sync_export_info record the exception is linked to.
SYNC_EXPORT_INFO	Contains information regarding export actions.	
	ID	Primary key.
	SYNC_END_CHECKPOINT	The name of the checkpoint created after the export.
	STATUS_RECORDS	1:n join field with the status records generated for the export.
	SUMMARY	Text summary.
	SYNC_EXCEPTIONS	1:n join field with the sync_exception records generated for the export.
	SYNC_METADATA	n:1 join field with the sync_metadata record of the export mapping.
STATUS_RECORD	Contains timestamps and mapping id's for export actions.	
	COMMIT_TIMESTAMP	The timestamp (also serves as primary key).
	MAPPING_ID	The mapping id for the action (always present for replications).
	OLD_CHECKPOINT	The name of the checkpoint created prior to the action.
	SYNC_EXPORT_INFO	n:1 join field with the sync_export_info record of the export.
SYNC_METADATA	Contains general information generated for an export mapping.	
	ID	Primary key.
	MAPPING_NAME	The name of the mapping.
	VIEW_NAME	The name of the Smallworld view used in the mapping.
	ALTERNATIVE_NAME	The name of the alternative used in the mapping.
	SYNC_EXPORT_INFOS	1:n join field with the sync_export_info records for the mapping.
SYNC_MAPPING_INFO	Contains general characteristics of mappings for future exports.	
	ID	Primary key.
	SYNC_CHECKPOINT	The name of the checkpoint generated for the mapping.
	TRANSFER_MODE	The type of export.
	MAX_ORA_ROWSCN	The maximum System Change Number for an Oracle row if existent.
	COLUMNS_TO_REFRESH	The column to be refreshed.

5.2.2 Oracle tables

Metadata about the synchronization is kept in Oracle in a set of tables. The information concerns various mappings: geometry, map, enumerator, dimensioning and manifold.

Other pieces of information are also stored, like replication/synchronization timestamps and spatial index names.

Below you can find a complete description of these tables.



SYNC_CONTAINED_GEOM	Contains the layers (combination of table and geometry field) which actually contain geometries. This table is populated during replications/synchronization.	
	ID	Primary key.
	MAP_TYPE_ID	The map-type this layer belongs to.
	TABLE_NAME	Name of the table.
	FIELD_NAME	Name of the field.
	VIEW_NAME	Name of the view through which this layer should be accessed.
	GEOM_TYPE	The numeric geometry type as used by Oracle Spatial. Used for creating the spatial indexes.
SYNC_DIMENSIONING_METADATA	Contains information regarding dimensioning mapping.	
	ID	Primary key.
	TABLE_NAME	The name of the table the dimensioning is mapped to.
	SW_DIM_NAME	The Smallworld name of the

		dimensioning.
	LINES_COLUMN	The column in the dimensioning table that holds line geometries.
	SYMBOLS_COLUMN	The column in the dimensioning table that holds symbols.
	SYMBOL_NAMES_COLUMN	The column in the dimensioning table that holds symbol names.
	LABELS_COLUMN	The column in the dimensioning table that holds label geometries.
	LABEL_TEXTS_COLUMN	The column in the dimensioning table that holds label texts.
SYNC_ENUMERATORS	Contains the names of the enumerators.	
	ENUMERATOR_NAME	Name of enumerator table.
	USED_BY_TABLE	Name of the table which uses the enumerator.
	USED_BY_FIELD	Name of the field which uses the enumerator.
SYNC_EXTERNAL_NAMES	Contains the external names of all the tables and fields.	
	TABLE_NAME	Name of the table.
	FIELD_NAME	Name of the field, this field is empty when the EXTERNAL_NAME field indicates an external name of a table.
	EXTERNAL_NAME	The external name of the table or field.
	LOCALE	Contains the locale of the external name. Currently unused.
SYNC_MANIFOLD_TO_NETWORK	Contains the manifold to network mappings.	
	ID	The Smallworld ID of the manifold. Primary key.
	MANIFOLD	The name of the Smallworld manifold.
	NETWORK	The name of the Oracle network.
SYNC_MAP	Contains all existing maps in the current dataset.	
	ID	The map-ID.
	MAP_TYPE_ID	The map-type of the map.
	BOUNDS	The boundary of the world as defined in Smallworld.
	WORLD_ID	The world ID (as defined in Smallworld).
	USED_BOUNDS	The actual used bounds of the map. This is calculated at the end of replication/synchronization.
	UNIT_FACTOR	The scaling factor for the measurement unit.
SYNC_MAP_TYPE	Contains the data of the universes.	
	ID	The map-type ID.
	UNIVERSE_ID	The ID of the Smallworld Universe.
	SRID	The Oracle Spatial Reference ID of the map-type.
	SCALING_FACTOR	The scaling factor relative to mm of

		non-earth coordinate systems.
	SCALING_UNIT	The scaling unit (usually 1.00).
	NAME	The name of the map-type (matches the name of the universe as defined in Smallworld).
	MULTIUNIT	Mentions if the map type is multiunit (0/1).
	VIEW_PREFIX	
SYNC_OWNER	Contains information about which tables are owners of internal worlds.	
	ID	Primary key.
	MAP_TYPE_ID	The map-type of the internal world.
	TABLE_NAME	Name of the table which has an internal world.
	MAP_ID_FIELD_NAME	Name of the field where the map-ID of the internal world is stored.
	CONTAINED_MAP_NAME	Name of the internal world-type.
SYNC_RWO_ID_MAPPING	Contains the RWO_IDs used in other tables.	
	RWO_ID_1	First element of the RWO_ID.
	RWO_ID_2	Second element of the RWO_ID.
	RWO_ID_3	Third element of the RWO_ID. On this field foreign key constraints are defined.
SYNC_SDO_INDEXES	Contains information about the spatial indexes created on the layers.	
	TABLE_NAME	Name of the table.
	COLUMN_NAME	Name of the geometry field.
	SDO_INDEX_NAME	Name of the spatial index.
	SDO_COLUMN_NAME	Name of the field on which the index is created. For geometry fields which contain geometries in different coordinate systems this field will contain the function call that is used to create the function based spatial index.
	SRID	The SRID used for this spatial index.
	MULTIMAP	Indicates if this layer contains geometries in different coordinate systems.
SYNC_STATUS	Status of the replication/synchronization process.	
	CONFIGURATION_NAME	Name of the configuration.
	TIMESTAMP	Time of the last commit.

5.3 Error descriptions

Unable to transform record for synchronization

This errors are raised when trying to transform the data from smallworld datastore format to flat format suitable for data marshaling. Usually the cause for this errors is data integrity or malformed geometries.

Could not extract geometry owning world

This kind of errors is raised when trying to insert a geometry into an Oracle table but the world in which the geometry exists can not be extracted from Smallworld.

The coordinates defining an arc are not distinct

The coordinates defining a Smallworld arc are not 3 distinct points.

Unable to update record due to missing database record.

A record that changed in Smallworld is missing in the Oracle table.

ORA-00001: unique constraint (xxxxx) violated

This error type happens when two or more records share the same primary key(s) or, in case of dataless record, they have the same 3rd component of their rwo_id key field.The error message has the form "<scheme>.<contraint>" (see http://www.cryer.co.uk/brian/oracle/ORA00001.htm). The scheme is the database scheme in which the table is located of which the contraint is violated, the scheme is also known as tablespace.

ORA-01461: can bind a LONG value only for insert into a LONG column

This kind of errors is raised when trying to insert a record in Oracle that has a filed of type string but the string value has more than 4000 characters.

ORA-00nnn: message

These messages are self descriptive.

Impossible to setup working view!

This message indicates that *SyncManager* is not able to create/use an alternative in the Smallworld database that is suitable for synchronization. Most likely this is caused by problems with merging the alternative. The solution is to create a new alternative, align it, or when you need the incremental synchronization, merge it manually.

ORA-01086: savepoint '<savepoint_name>' never established

This error is caused because the UNDO tablespace (of Oracle) is not large enough and this results in an error message when every 1000 records get commited using the subtransaction feature of Oracle (savepoints). This results in a large quantity of error messages and the max of 1000 or 10.000 is easily made, then the sync crashes.

There is a workaround for this problem, by changing the settings of the UNDO tablespace, so it can grow unlimited:

```
ALTER DATABASE swosw DATAFILE 'D:\ORACLE\PRODUCT\10.2.0\ORADATA\SWOSW
\UNDOTBS1.DBF' AUTOEXTEND ON NEXT 100M UNLIMITED;
```

Get your undo tablespace datafile:

```
SELECT file_name
FROM dba_data_files
WHERE tablespace_name LIKE 'UNDO%'
```

5.4 Known limitations

SyncManager offers an interface for data exchange between complex systems. Partly due to this, it has some limitations. The limitations that are known to us are listed here.

Synchronization of character fields limited to 4000 chars

When a field contains more than 4000 characters, it cannot be synchronized. The system will return the message ORA-01461 (see error messages 62). The only workaround for this is to remove the field from the mapping, or find a way to compress the content.

Coordinates

Problems may arise due to problematic or erroneous coordinates in Smallworld, that may lead for example to an error message like "The coordinates defining an arc are not distinct" or "the coordinates defining an arc are collinear" Some of them are a combination of a problem between scaling and arcs in smallworld. There are also known limitations through rounding errors by going from mm in Smallworld to meters in oracle e.g. Make sure your source data is valid.

Part VI

6 Release notes

Version 2.2.1-41100

Changes to version 2.2.0-41000:

SYNC-2804,3279 - Used bounds not updated if geometry is in two worlds

If geometries reside in a world other than main world, or in other words, geometries with map type other than 1, the bounds never got updated. This means that the value of the column USED_BOUNDS in the Oracle table SYNC_MAP was always empty even though the supplied command line utility was used to calculate the bounds. This lead to problems with the scale of those map types. This bug is fixed in this version.
Affected versions are: 1.1.6, 2.0, 2.2.0.

SYNC-3039, 3041- Deadlock while trying to disable constraints

In certain cases synchronization could lead to the exception "ORA-04020: deadlock detected while trying to lock object x.x"
This was caused by the way SyncManager removed constraints from the tables in a scheme. SyncManager disabled constraints in ALL tables in a schema, also leading to problems with other non-SM tables used by other applications at the time. SyncManager now disables only constraints for tables that it uses. And it's done right before the data for that table being transferred from Smallworld to Oracle. After the syncing process is done, all constraints that has been disabled during the syncing process will be re-enabled back.
A patch for 2.0.0 is released. The problem is fixed in this version.

SYNC-3061, 3063 - Extremely low syncing speed due to missing rwo id's

In certain cases the synchronization, even on small tables, was extremely low. In some cases this could take 20 hours. It appeared that this was caused by geometries with a value 0 for the field rwo_id. Fixing this problem will solve the problem.

SYNC-3091 - Partly specified configuration can be saved, leading to traceback

When setting the option Dimensioning in the Smallworld Preferences from "No" to "Yes", e.g. the name of the ace could left to be left blank.
Lateron this results in a traceback, as the acename is evaluated as "", the empty string. This is fixed. Now all the configuration have to be filled before the configuration can be saved, otherwise the GUI will give warning to the user.

SYNC-3390,3424 - Support for Oracle RAC connection: new expert mode for connection string in the Oracle Preferences menu

For connections to Oracle RAC, a more advanced connection string is needed. In this release the Oracle Preference menu 30 offers two modes. The existing (simple) mode that creates the connection string and an Expert mode that allows you to add more parameters to the connection string.

SYNC-3398 - Unknown coordinate systems

In version 2.2.0 it could happen that the automatic mapping of coordinate systems between Smallworld and Oracle failed with the error message *"java.io. FileNotFoundException: coordinate_systems.xml (The system cannot find the file specified)"*. The automated mapping was designed to fall back on an xml-file with mappings between coordinate systems, but which was removed in 2.2.0. In this version all mappings between coordinate systems that can not be created automatically, have to be done manually using the interface as described in section Map Types Page 34.

SYNC-3414 - Incompatible licences

A bug was found with the license mechanism for some systems where the encryption was effected, therefore for some customers new licenses might be required.

SYNC-3425 - Error when adding table that owns an internal world

It is reported that tables that own internal worlds that do not have a name leads to an error in SyncManager. There is no software changed for this. Customers should add a name to the worlds of tables to be synchronized.

SYNC-3426 - Error building spatial index

In prior versions of SyncManager if the geometry column metadata was registered as being multi-line and the actual geometry entries contained also line geoms, the spatial index build routine would fail. This was fixed, but when the spatial index would fail due to unknown reasons, the routine would try to build an normal index instead of a spatial index. This is fixed. If the spatial index build routine fails because of the data itself then the error that caused the failure will be displayed.

SYNC-3428 - Traceback when maximum number of records is exceeded

When the maximum number of errors was reached, SyncManager faild with the error message "Object a sync_manager(syncmanager) does not understand message set_operation_status() with arguments: False"
This is now fixed.

Version 2.2.0-41000

Changes to version 2.0.M3:

SYNC- Improved User Interface of the Mapping Editor 25

The Mappings Definitions Menu has disappeared and it's functionality has been replaced by that of the Contextual Menu 31. The Enumerator Mapping Page 32, Network Mapping Page 33 and Coordinate Systems Page 34 have also changed.

SYNC- Changed User Interface for the Update Datamodel function

The **Update DD** button has disappeared from all the pages and it's functionality is now replaced by the **Update Datamodel** action in the Datamodel menu.

SYNC- Changed User Interface for Full Sync Option

The management of the **Full Sync** option is now done through the switch in each collection's individual Table Info Page 37, as well as the **Switching Full Sync** sets **Truncate** to **ON** option in the Contextual Menu 31.

SYNC-3043 - Support for Smallworld Core Spatial Technology 4.1

This release now supports GIS core 4.1. This means it now also supports the use of 64 bits UVA.

SYNC-3066 - Removed default spatial index tablespace option from Oracle preferences page.

The default spatial index tablespace cfg options were removed because the provided value was not used and it is not possible to use them.

SYNC- Removed spatial tablespace anchors on geometry column mapping info pages.

SYNC- List of Oracle coordinate systems closes itself when dbl-clicking an item.

When in the Map Types page 34 a Map type is selected with a double click, the dialog will be closed.

SYNC-2814 - SyncManager Oracle connection failed due to non-updated text fields

A reported issue on connection problems revealed a flaw in the interface where the text fields were not saving the last entered value unless tab or enter was pressed. The solution to this is that text fields are saving the entered value when the focus is lost or tab/enter is pressed.

SYNC-2857 - SyncUtils crashes instead of errormessage if database parameters are faulty

When connecting to the database with faulty database parameters the following exception trace may occur:

```
Exception in thread "main" java.lang.NullPointerException
at com.realworldwide.syncmanager.postprocess.oracle.OracleSyncUtil.handleAction
(OracleSyncUtil.java:311)
```

```
at com.realworldwide.syncmanager.postprocess.oracle.OracleSyncUtil.run(OracleSyncUtil.
java:352)
at com.realworldwide.syncmanager.postprocess.oracle.OracleSyncUtil.main(OracleSyncUtil.
java:42)
```

The problem was caused by the program that keeps trying to run even though it doesn't have a valid Oracle connection. The user usually cannot see this due to default log4j logging parameters
This problem is solved now. An enhancement request for improvement of the logging is issued.

SYNC-2870 - Referential constraints are not created

This bug affected only the configuration files created with version 1.x.x of the product.
cause: when parsing an old configuration file, the constraints were identified as being already created although the datamodel was not created yet.
This is now fixed. the constraint are properly created when applying the datamodel and their old "created" status is not kept in the xml file anymore.

SYNC-2888 - Errors in Path Table

Values stored in the path_type column of the PATH table were inconsistent across full exports.
This has been fixed.

SYNC-2903 - Unable to save the configuration

When a mapping configuration was corrupted somehow, the program did not respond to this situation properly.
This is now fixed. The mapping parser now ignores and/or discards the corrupted mappings.

SYNC-2928 - DataModel Update

SYNC-3018 - Configuration can't be saved because of dimensioning

SYNC-3029 Non standard internal world owners not being properly identified.

This problem has been solved by introducing a new format for the definition of worlds and world owners. See for a description **SYNC-3071 - Change in the definition of worlds.**

SYNC-3030 - Key fields should not be available for remove operation

When removing field mappings from a collection mapping, the key fields are no longer available for selection.

SYNC-3031 - Changing the dimensioning ACE doesn't validate the scale used.

When **include dimensioning** in the Smallworld Preferences 29 is set to **yes**, an ACE, Style Group and Scale can be selected. Since the Scale and Style are dependent on the ACE, now the user has to re-select them if the ACE is changed.

SYNC-3036 - Oracle constraints name change causes the update datamodel to crash.

When changing a constraint name, the update operation failed because the engine reports that there is already a constraint defined between the tables. In this version it'll only create the constraint if it does not exist yet. But if there's already a constraint and

the name is different, it will rename it.

Unresolved issues:

SYNC-3075 - Help button does not respond

The Help button does not provide help. This will be fixed in version 2.3.0.

Version 2.0.M3

Changes to version 1.1.6:

SYNC - Improved User Interface of the Mapping Editor 25

In this release the User Interface of the Mapping editor is changed substantially. The new interface is focused on navigation between mappings, related mappings, provides more information for mapping elements, and introduces the ability to search through most of the configuration data.

SYNC - Change of pathnames of the Smallworld modules

The environment configuration file **sync_manager_configuration.xml** is relocated from:

SyncManagerInstallDir\sync_manager\src\sync_manager_src\resources\base\data

to:

SyncManagerInstallDir\roos_sync_manager\modules\src\sync_manager\resources\base\data

SYNC - Added the environment variable SYNC_HOME

The value of this new variable should point to the location on disk of the **roos_sync_manager** module. See Environment settings 18.

SYNC-2769 - Certain fields of records are not being updated (joins)

Joins implemented with intermediate tables (1:x) failed to be exported. This is now fixed.

SYNC-2879 - Re-export Configuration

There was no way to re-export all the collections without dropping the Oracle datamodel and opening each mapping (which might be hundreds) to force re-export.
In this version there is now an option in the Configuration menu that allows full re-export of a configuration. The menu item is named **Re-Export Configuration**

SYNC-2883 - Datamodel update fails

When removing a field from a column mapping and initiating a datamodel update, this might fail with the error Update datamodel failed.

The GUI engine properly identifies the mappings that require their tables being created in Oracle and for those it first drops any existing table before creating the new one. This issue was describing the case when removing a field mapping caused failure when applying the datamodel changes. In the fixed version, only saving the configuration is required. Even if the "Apply Datamodel Changes" is enabled, due to other changes, the removal of the field mapping is not propagated to Oracle. The only time when field removals are propagated, is when they have been removed from the Smallworld datamodel.

SYNC-2876 - Can't deactivate a re-export even if a synchronization happened

In some cases the re-export of full sync status was not set properly. This is now fixed. It can only be changed for collection mappings that have already been replicated.

SYNC-2875 - Re-export is not taken into account

When changing only the Full sync status of a collection, the configuration could not be saved and used.
This is now fixed and full sync status is properly propagated.

SYNC-2873 - Change for tablespace are not taken into account

Changing a tablespace was not always properly saved in the configuration. This is fixed and changes to TABLESPACE name are propagated to Oracle as well.

SYNC-2872 - Problem on interface due to unverified Oracle connection when we open an existing configuration

The change of some parameters, like the lists of tablespaces, require a validated connection to Oracle. This was not done automatically when opening a configuration, resulting e.g. in empty lists.
This is now fixed. The Oracle connection is verified when the configuration is first opened.

SYNC-2877 - Changed operation of truncation

In the previous release, for a re-export you had to truncate table manually. There was a message to inform.
In this release there is a new option in the popup menu that controls how truncate works. See here 31.
The option is named **Switching Full Sync sets Truncate ON**, and is by default turned off. This means that when switching the **Full Sync** to yes for a collection, next synchronisation the collection will be refreshed, i.e. all records will be sent as updates and missing records will be intercepted and treated as inserts. If the **Switching Full Sync sets Truncate ON** is YES, a full synced collection will be replicated, i.e. all records will be sent as inserts.
The "Switching Full Sync sets Truncate ON" change does not affect earlier changes made to collection mappings.

SYNC-3071 Change in the definition of worlds

Release 2 contains a change in the way the world owners are specified. This change is not backwards compatible to the constants defined in prior releases, so you have to change the settings as described in the section Internal world owner list 19.

realworldsystems

2009